9000A-8086

Interface Pod

Instruction Manual

FLUKE®

Digitized by the Internet Archive
in 2025

https://archive.org/details/manualsplus_09174

NOTE

This manual documents the Model 9000A-8086 and its assemblies at the revision levels identified in Section 7. If your instrument contains assemblies with different revision letters, it will be necessary for you to either update or backdate this manual. Refer to the supplemental change/errata sheet for newer assemblies, or to the backdating information in Section 7 for older assemblies.

9000A-8086
Interface Pod

Instruction Manual

P/N 649426
SEPTEMBER 1982
©1982, John Fluke Mfg. Co., Inc., all rights reserved. Litho in U.S.A.

FLUKE®

WARRANTY

Notwithstanding any provision of any agreement the following warranty is exclusive:

The JOHN FLUKE MFG. CO., INC., warrants each instrument it manufactures to be free from defects in material and workmanship under normal use and service for the period of 1 year from date of purchase. This warranty extends only to the original purchaser. This warranty shall not apply to fuses, disposable batteries (rechargeable type batteries are warranted for 90 days), or any product or parts which have been subject to misuse, neglect, accident, or abnormal conditions of operations.

In the event of failure of a product covered by this warranty, John Fluke Mfg. Co., Inc., will repair and calibrate an instrument returned to an authorized Service Facility within 1 year of the original purchase; provided the warrantor's examination discloses to its satisfaction that the product was defective. The warrantor may, at its option, replace the product in lieu of repair. With regard to any instrument returned within 1 year of the original purchase, said repairs or replacement will be made without charge. If the failure has been caused by misuse, neglect, accident, or abnormal conditions of operations, repairs will be billed at a nominal cost. In such case, an estimate will be submitted before work is started, if requested.

THE FOREGOING WARRANTY IS IN LIEU OF ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS, OR ADEQUACY FOR ANY PARTICULAR PURPOSE OR USE. JOHN FLUKE MFG. CO., INC., SHALL NOT BE LIABLE FOR ANY SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WHETHER IN CONTRACT, TORT, OR OTHERWISE.

If any failure occurs, the following steps should be taken:

1. Notify the JOHN FLUKE MFG. CO., INC., or nearest Service facility, giving full details of the difficulty, and include the model number, type number, and serial number. On receipt of this information, service data, or shipping instructions will be forwarded to you.

2. On receipt of the shipping instructions, forward the instrument, transportation prepaid. Repairs will be made at the Service Facility and the instrument returned, transportation prepaid.

SHIPPING TO MANUFACTURER FOR REPAIR OR ADJUSTMENT

All shipments of JOHN FLUKE MFG. CO., INC., instruments should be made via United Parcel Service or "Best Way"* prepaid. The instrument should be shipped in the original packing carton; or if it is not available, use any suitable container that is rigid and of adequate size. If a substitute container is used, the instrument should be wrapped in paper and surrounded with at least four inches of excelsior or similar shock-absorbing material.

CLAIM FOR DAMAGE IN SHIPMENT TO ORIGINAL PURCHASER

The instrument should be thoroughly inspected immediately upon original delivery to purchaser. All material in the container should be checked against the enclosed packing list. The manufacturer will not be responsible for shortages against the packing sheet unless notified immediately. If the instrument is damaged in any way, a claim should be filed with the carrier immediately. (To obtain a quotation to repair shipment damage, contact the nearest Fluke Technical Center.) Final claim and negotiations with the carrier must be completed by the customer.

The JOHN FLUKE MFG. CO., INC, will be happy to answer all applications or use questions, which will enhance your use of this instrument. Please address your requests or correspondence to: JOHN FLUKE MFG. CO., INC., P.O. BOX C9090, EVERETT, WASHINGTON 98206, ATTN: Sales Dept. For European Customers: Fluke (Holland) B.V., P.O. Box 5053, 5004 EB, Tilburg, The Netherlands.

*For European customers, Air Freight prepaid.

John Fluke Mfg. Co., Inc., P.O. Box C9090, Everett, Washington 98206

Rev. 6/81

Table of Contents

SECTION	TITLE	PAGE
1	**INTRODUCTION**	**1-1**
	1-1. PURPOSE OF INTERFACE POD	1-1
	1-2. DESCRIPTION OF POD	1-1
	1-3. SPECIFICATIONS	1-2
2	**INSTALLATION AND SELF TEST**	**2-1**
	2-1. INTRODUCTION	2-1
	2-2. PERFORMING THE POD SELF TEST	2-1
	2-3. CONNECTING THE POD TO THE UUT	2-1
3	**MICROCOMPUTER DATA**	**3-1**
	3-1. INTRODUCTION	3-1
	3-2. MICROCOMPUTER SIGNALS	3-1
4	**OPERATING CHARACTERISTICS**	**4-1**
	4-1. INTRODUCTION	4-1
	4-2. GETTING STARTED	4-1
	4-3. ADDRESS SPACE ASSIGNMENT	4-2
	4-4. Introduction	4-2
	4-5. Special Addresses	4-3
	4-6. Role of 8086 Segment Registers in Memory Addressing	4-3
	4-7. Demultiplexing Memory Addresses with the A16/S3 and A17/S4 Lines	4-4
	4-8. Introduction	4-4
	4-10. Status Line Bit Assignments	4-5
	4-11. User-Enableable Lines	4-6
	4-12. Status Lines Unique to the Pod	4-6
	4-17. Forcing Lines	4-7
	4-18. Interrupt Lines	4-8
	4-19. User-Writeable Lines	4-8
	4-22. Control Line Bit Assignments	4-9
	4-23. SPECIAL FUNCTIONS UNIQUE TO THE 8086 POD	4-9
	4-24. Introduction	4-9
	4-25. Quick Looping Read or Write	4-12
	4-26. Quick RAM Test	4-12
	4-27. Quick ROM Test	4-13
	4-28. RUN UUT MODE	4-14
	4-29. Run UUT Default Execution Address	4-14
	4-30. Specifying Execution Address	4-14

(continued on page ii)

TABLE OF CONTENTS, *continued*

SECTION	TITLE	PAGE
	4-31. Specifying Segment Register Contents Before Execution	4-16
	4-32. DEFAULT ADDRESSES FOR LEARN AND BUS TEST	4-16
	4-33. Learn Operation Default Address	4-16
	4-34. Default Address for Data Line Testing for Bus Test	4-16
	4-35. INTERRUPT HANDLING	4-16
	4-36. Introduction	4-16
	4-37. Using Special Addresses to Read Interrupt Type and Cascade Address	4-17
	4-38. Forcing the Execution of the Interrupt Acknowledge Routine	4-17
	4-39. PROBE AND SCOPE SYNCHRONIZATION MODES	4-18
	4-40. PROBLEMS DUE TO A MARGINAL UUT	4-19
	4-41. Introduction	4-19
	4-42. UUT Operating Speed and Memory Access	4-19
	4-43. UUT Noise Levels	4-19
	4-44. Bus Loading	4-19
	4-45. Clock Loading	4-19
	4-46. POD DRIVE CAPABILITY	4-19
	4-47. LOW UUT POWER DETECTION	4-20
5	**THEORY OF OPERATION**	**5-1**
	5-1. INTRODUCTION	5-1
	5-2. GENERAL POD OPERATION	5-1
	5-3. Processor Section	5-1
	5-4. Timing and Control Section	5-4
	5-5. Interface Section	5-4
	5-6. DETAILED BLOCK DIAGRAM DESCRIPTION	5-5
	5-7. Detailed Description of Processor Section	5-5
	5-8. Detailed Description of Timing and Control Section	5-8
	5-9. Interface Section	5-11
6	**TROUBLESHOOTING**	**6-1**
	6-1. INTRODUCTION	6-1
	6-2. DETERMINING WHETHER THE POD IS DEFECTIVE OR INOPERATIVE	6-2
	6-3. TROUBLESHOOTING A DEFECTIVE POD	6-2
	6-4. Preparation for Troubleshooting a Defective Pod	6-3
	6-5. Recreating the Enhanced Self Test Routines	6-5
	6-6. Recreating the Standard Self Test Routines	6-6
	6-7. TROUBLESHOOTING AN INOPERATIVE POD	6-7
	6-8. Introduction	6-7
	6-9. Procedure for Troubleshooting an Inoperative Pod	6-7
	6-10. EXTENDED TROUBLESHOOTING PROCEDURES	6-11
	6-11. Partially Checked Circuits	6-11
	6-12. Timing and Noise Problems	6-12
	6-13. DISASSEMBLY	6-12
7	**LIST OF REPLACEABLE PARTS**	**7-1**
	7-1. INTRODUCTION	7-1
	7-2. HOW TO OBTAIN PARTS	7-1
	7-3. MANUAL CHANGE AND BACKDATING INFORMATION	7-2

(continued on page

TABLE OF CONTENTS, *continued*

SECTION	TITLE	PAGE
8	**SCHEMATIC DIAGRAMS**	**8-1**
	TABLE OF CONTENTS	8-1
	INDEX	**8-12**

List of Tables

TABLE	TITLE	PAGE
1-1.	8086 Pod Specifications	1-4
3-1.	Microprocessor Signals	3-1
4-1.	Address Space Assignment	4-3
4-2.	Status and Control Line Bit Assignments	4-5
4-3.	Special Functions Unique to the 8086 Pod	4-10
4-4.	Special Addresses for Run UUT Mode	4-15
4-5.	Special Addresses for Interrupt Handling	4-18
6-1.	Required Test Equipment for Pod Troubleshooting	6-2
6-2.	Standard Self Test Failure Codes	6-3
6-3.	Enhanced Self Test Failure Codes	6-6
6-4.	Pod Device Addresses	6-9
6-5.	Bit Definitions for Selected Pod Addresses	6-10
6-6.	Pod Ribbon Cable Lines Partially Checked in Pod Self Test	6-11

List of Illustrations

FIGURE	TITLE	PAGE
1-1.	Communication Between the Troubleshooter, the Pod, and the UUT	1-3
1-2.	External Features of the 8086 Interface Pod	1-3
2-1.	Connection of Interface Pod to Troubleshooter	2-2
3-1.	8086 Pin Assignments	3-5
5-1.	8086 Interface Pod General Block Diagram	5-2
5-2.	8086 Interface Pod Detailed Block Diagram	5-6
5-3.	Handshake Signals	5-8
5-4.	8086 Signal Timing Relationships	5-9
5-5.	The Request/Grant Circuitry	5-12
6-1.	Troubleshooting a Defective Pod	6-5
6-2.	Troubleshooting an Inoperative Pod	6-8

Section 1
Introduction

NOTE

It is assumed that the user of this manual is familiar with the basic operation of one of the 9000 Series Micro System Troubleshooters as described in the 9000 Series Operator manuals.

1-1. PURPOSE OF INTERFACE POD

The purpose of the 9000A-8086 Interface Pod (hereafter referred to as the pod) is to interface any 9000 Series Micro System Troubleshooter (hereafter referred to as the troubleshooter) to equipment employing an 8086 microprocessor.

The troubleshooter is designed to service printed circuit boards, instruments, and systems employing microprocessors. The architecture of the troubleshooter is general in nature, and is designed to accommodate devices with up to 32 address lines and 32 data lines. The pod adapts the general purpose architecture of the troubleshooter to a specific microprocessor or microprocessor family. The pod adapts such microprocessor-specific functions as pin layout, status/control functions, interrupt handling, timing, and memory and I/O addressing.

1-2. DESCRIPTION OF POD

Figure 1-1 shows the communication between the pod, the troubleshooter, and the unit-under-test (hereafter referred to as the UUT). The pod connects to the troubleshooter through a shielded 24-conductor cable. The pod connects to the UUT through the microprocessor socket, which gives the troubleshooter access to all system components which normally communicate with the microprocessor.

The external features of the pod are shown in Figure 1-2. The pod consists of a pair of printed circuit board assemblies mounted within a break-resistant case. The pod contains an 8086-1 microprocessor along with the supporting hardware and control software that is required to do the following:

1. Perform handshaking with the troubleshooter.
2. Receive and execute commands from the troubleshooter.
3. Report UUT status to the troubleshooter.
4. Allow the UUT microprocessor to operate with the UUT.

The troubleshooter supplies operating power (+5V,+12V and -5V) for the pod. The UUT provides the external clock signal required by the pod for operation. Using the UUT

clock signal allows the troubleshooter and pod to function at the designed operating speed of the UUT.

Logic level detection circuits are provided on each line to the UUT. These circuits allow detection of bus shorts, stuck-high or stuck-low conditions, and any bus drive conflict (two or more drivers attempting to drive the same bus line).

Over-voltage protection circuits are also provided on each line to the UUT. These circuits guard against pod damage which could result from the following:

1. Incorrectly inserting the ribbon cable plug in the UUT microprocessor socket.

2. UUT faults which place potentially-damaging voltages on the UUT microprocessor socket.

The over-voltage protection circuits guard against voltages of +12 to -7 volts on any one pin. Multiple faults, especially of long duration, may cause pod damage.

A power level sensing circuit constantly monitors the voltage level of the UUT power supply (+5V). If UUT power rises above or drops below an acceptable level the pod notifies the troubleshooter of the power fail condition.

A self test socket provided on the pod enables the troubleshooter to check pod operation. The self test socket is a 40-pin zero-insertion force type socket. The ribbon cable plug must be connected to the self test socket during self test operation. The ribbon cable plug should also be inserted into this socket when the pod is not in use to provide protection for the plug.

1-3. SPECIFICATIONS

Specifications for the pod are listed in Table 1-1.

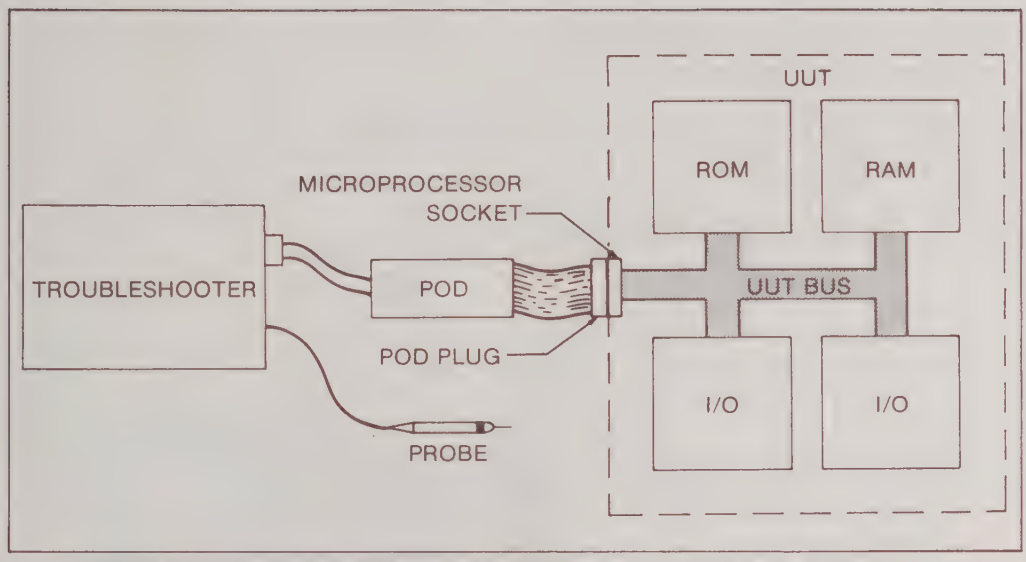


Figure 1-1. Communication Between the Troubleshooter, the Pod, and the UUT

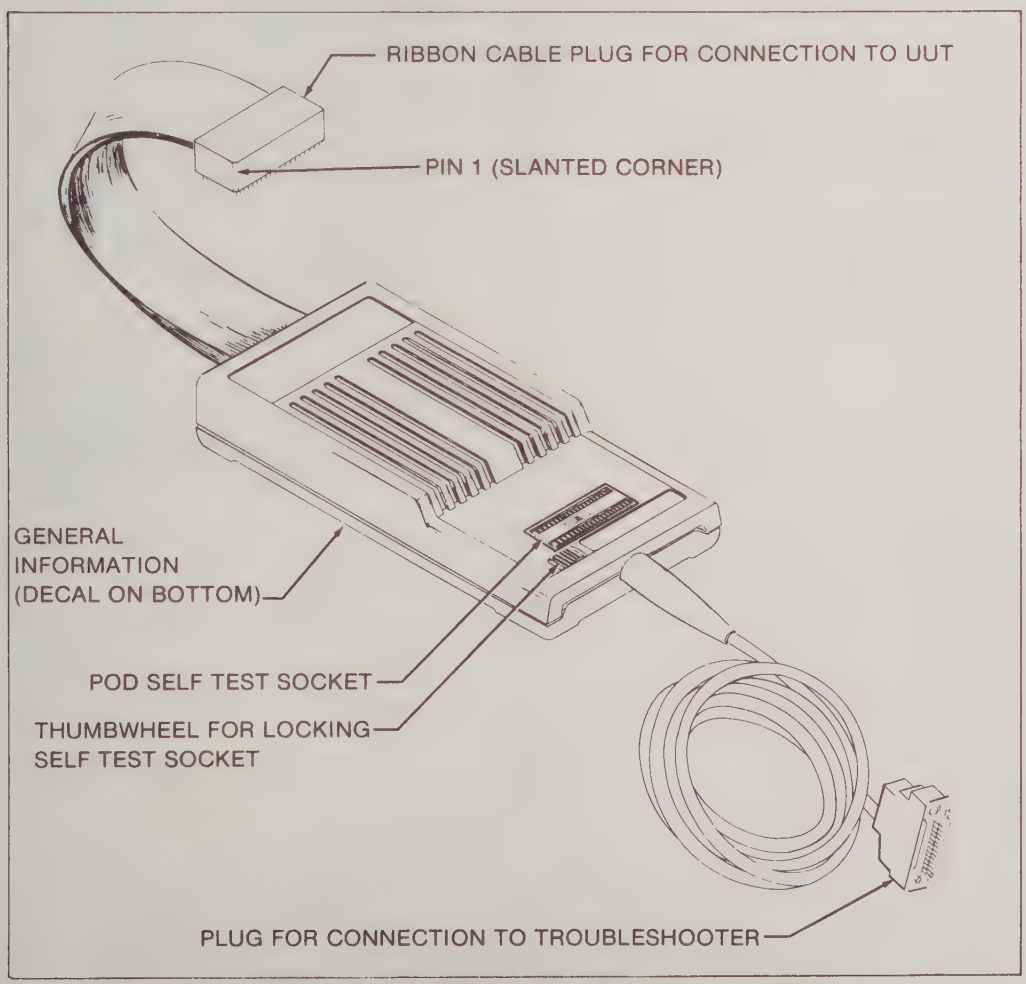


Figure 1-2. External Features of the 8086 Interface Pod

Table 1-1. 8086 Pod Specifications

ELECTRICAL PERFORMANCE

Power Dissipation	10.0 watts maximum
Maximum External Voltage	-7 to +12 volts may be applied between ground and any ribbon cable plug pin continuously.

MICROPROCESSOR SIGNALS

Clock Input Low	0V min., +0.45V max.
Clock Input High	+4.4V min., +5.0V max.
Input Low Voltage	0V min., +0.8V max.
Input High Voltage	+2.0V min., +5.0V max.
Output Low Voltage	+0.45V max. with Iol = 2.5 mA
Output High Voltage	+2.4V min. with Ioh = 400 μA
Tristate Output Leakage Current	+20/-100 μA typical
High Level Input Current	20 uA (typ.) with Vih = +2.7V
Low Level Input Current	-400 uA (typ.) with Vil = +0.4V

TIMING CHARACTERISTICS

Maximum Clock Frequency	
MAX. MODE	8 MHz
MIN. MODE	10 MHz
Added Delays to 8086 Signals	
LOW-TO-HIGH TRANSITIONS	20 ns typical
HIGH-TO-LOW TRANSITIONS	24 ns typical

UUT POWER DETECTION

Detection of Low Vcc Fault	Vcc < +4.5V
Detection of High Vcc Fault	Vcc > +5.5V

GENERAL

Size	5.7 cm H x 14.5 cm W x 27.1 cm L (2.2 in H x 5.7 in W x 10.7 in L)
Weight	1.4 kg (3.1 lbs)
Environment	
STORAGE	-40°C to +70°C, RH*<95%
OPERATING	0°C to +40°C, RH*<95% +40°C to +50°C, RH*<75%
Protection Class 3	Relates solely to insulation or grounding defined in IEC 348.

*All relative humidity (RH) conditions are non-condensing.

Section 2
Installation and Self Test

2-1. INTRODUCTION

The procedures for performing the pod self test and connecting the pod to the troubleshooter and the UUT are given in the following paragraphs.

2-2. PERFORMING THE POD SELF TEST

To perform the pod self test, perform the following steps:

1. Remove power from the troubleshooter.

2. Open the pins of the pod self test socket by operating the adjacent thumbwheel. Insert the ribbon cable plug into the socket and close the socket using the thumbwheel.

3. Using the round shielded cable, connect the pod to the troubleshooter at the location shown in Figure 2-1. Secure the connector using the sliding collar.

4. Apply power to the troubleshooter.

5. Press the BUS TEST key to initiate the pod self test.

If the troubleshooter displays the message *POD SELF TEST 8086 OK*, then the pod is operating properly.

If the troubleshooter displays the message *POD SELF TEST 8086 FAIL xx*, the pod may not be operating properly. (The letters *xx* correspond to a failure code describing the error; the failure codes are listed in Section 6.) Make sure the pod ribbon cable plug is properly positioned in the self test socket and try the self test again.

For information about pod troubleshooting and repair, refer to Section 6.

2-3. CONNECTING THE POD TO THE UUT

WARNING

TO PREVENT POSSIBLE HAZARDS TO THE OPERATOR OR DAMAGE TO THE UUT, DISCONNECT ALL HIGH-VOLTAGE POWER SUPPLIES, THERMAL ELEMENTS, MOTORS, OR MECHANICAL ACTUATORS WHICH ARE CONTROLLED OR PROGRAMMED BY THE UUT MICROPROCESSOR BEFORE CONNECTING THE POD.

Connect the pod between the troubleshooter and the UUT as follows:

1. Be sure that power is removed from the UUT.

2. Disconnect UUT analog outputs or potentially hazardous UUT peripheral devices as described in the warning at the beginning of this section.

3. Disassemble the UUT to gain access to the UUT microprocessor socket. If the UUT microprocessor is still in the socket, remove the microprocessor.

4. Turn the pod self test socket thumbwheel to release the pod plug, and remove the pod plug from the self test socket.

5. Insert the pod plug into the UUT microprocessor socket. Make sure the slanted corner of the pod plug is aligned with pin 1 of the UUT microprocessor socket.

6. Reassemble the UUT using extender boards if necessary.

CAUTION

Be sure that troubleshooter power is on before turning UUT power on in order to activate pod protection circuits.

7. Apply power to the troubleshooter and the UUT.

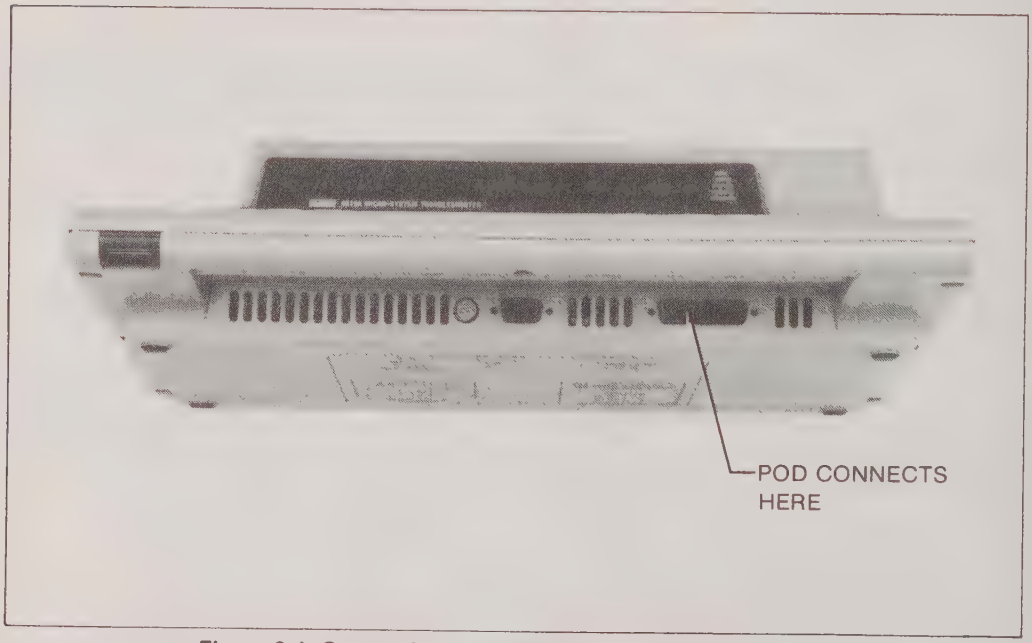


Figure 2-1. Connection of Interface Pod to Troubleshooter

Section 3
Microprocessor Data

3-1. INTRODUCTION

This section contains microprocessor data which may be useful during operation of the troubleshooter. This information includes descriptions of 8086 signals and pin assignments for both the minimum and maximum modes of operation.

3-2. MICROPROCESSOR SIGNALS

Table 3-1 lists all of the 8086 microprocessor signals and provides a brief description of each signal. Figure 3-1 shows the 8086 pin assignments.

Table 3-1. Microprocessor Signals

<table>
<tr><th>SIGNAL NAME</th><th>DESCRIPTION</th></tr>
<tr><td>AD0-AD15</td><td>ADDRESS/DATA LINES. These 16 input/output lines constitute 16 lines of the time-multiplexed memory address bus (during T1 of the bus cycle) and the data bus (during T2, T3, TW, and T4 of the bus cycle).

The AD0 line is used for byte selection in order to transfer a byte of data from memory onto the lower half of the data bus (this line is always low for even word accesses). For more information, refer to the section titled Address Space Assignment in Section 4.</td></tr>
<tr><td>A19/S6
A18/S5
A17/S4
A16/S3</td><td>ADDRESS/STATUS LINES. These four output lines constitute four lines of the time-multiplexed memory address bus (during T1 of the bus cycle) and four bus cycle status lines (during T2, T3, TW, and T4 of the bus cycle). S5 reflects the state of the interrupt enable flag. S6 is unused and always equals 0. Refer to the following table for status line coding of lines S3 and S4:

<table>
<tr><th>S4</th><th>S3</th><th>REGISTER USED</th></tr>
<tr><td>0</td><td>0</td><td>Extra data (ES)</td></tr>
<tr><td>0</td><td>1</td><td>Stack (SS)</td></tr>
<tr><td>1</td><td>0</td><td>Code (CS) or none</td></tr>
<tr><td>1</td><td>1</td><td>Data (DS)</td></tr>
</table>
</td></tr>
</table>

Table 3-1. Microprocessor Signals (cont)

SIGNAL NAME	DESCRIPTION
$\overline{BHE}$/S7	$\overline{BUS\ HIGH\ ENABLE}$/STATUS LINE. This output is typically made low (during T1 of the bus cycle) to transfer a byte of data from memory onto the upper half of the data bus (refer to Memory Addressing in Section 4 for more information). This line is equivalent to the AD0 line, but for the upper (odd) memory bank and the upper half of the data bus (D8-D15). S7 is a spare status bit whose value is undefined.
$\overline{RD}$	$\overline{READ}$ CONTROL LINE. This output is made low to notify external devices that the 8086 is ready to read data from the time multiplexed AD0-AD15 lines.
MN/$\overline{MX}$	MINIMUM/$\overline{MAXIMUM}$ LINE. This input allows the 8086 to be operated in the minimum mode or maximum mode. A +5V applied to this line selects the minimum mode in which the 8086 supports small, single-processor systems that consist of a few devices configured around a system bus. Ground applied to this line selects the maximum mode in which a bus controller, rather than the 8086, provides all bus control and command outputs (such as in a multi-processor system). In the maximum mode, the 8086 allows the lines previously delegated to bus control and command outputs to be redefined in order to support multi-processing functions.
$\overline{S2}$,$\overline{S1}$,$\overline{S0}$	STATUS LINES (maximum mode). When the 8086 is configured in the maximum mode, these output lines provide eight status signals for use by external devices. The status signals identify the type of bus cycle the 8086 is starting to execute. See the following table for status line coding.

$\overline{S2}$	$\overline{S1}$	$\overline{S0}$	TYPE OF BUS CYCLE
0	0	0	Interrupt Acknowledge
0	0	1	Read I/O
0	1	0	Write I/O
0	1	1	Halt
1	0	0	Instruction Fetch
1	0	1	Read Memory
1	1	0	Write Memory
1	1	1	Passive, no bus cycle

Table 3-1. Microprocessor Signals (cont)

SIGNAL NAME	DESCRIPTION
QS0,QS1	QUEUE STATUS LINES (maximum mode). When the 8086 is configured in the maximum mode, these output lines permit external monitoring of the internal instruction queue. This allows instruction set extension processing by a co-processor.
$\overline{\text{LOCK}}$	$\overline{\text{LOCK}}$ OUTPUT LINE (maximum mode). When the 8086 is configured in the maximum mode, this software-controlled output is used in conjunction with an external bus arbiter to guarantee exclusive access of a shared system bus for the duration of an instruction. During an interrupt acknowledge sequence, the Lock output is activated to signal bus arbiters in the system that the bus should not be accessed by any other processor.
$\overline{\text{RQ}}/\overline{\text{GT0}}$	$\overline{\text{REQUEST}}/\overline{\text{GRANT LINE}}$ (maximum mode). This input/output is used in multi-processor applications to request and grant bus usage, and is tied to the same line of the CPU. The request/grant sequence is a three-phase cycle: 1. Request Phase: the requesting device outputs a pulse on the $\overline{\text{RQ}}/\overline{\text{GT0}}$ line. 2. Grant Phase: the CPU outputs a pulse (onto this same line) at the end of either the current bus cycle or the idle clock period, indicating to the requesting device that it has floated the system bus, and is about to disconnect from the bus controller and enter a hold state. 3. Release Phase: the requesting device again outputs a pulse onto the same line to notify the CPU that it is ready to release the bus. The CPU regains bus access on the next clock cycle.
$\overline{\text{RQ}}/\overline{\text{GT1}}$	$\overline{\text{REQUEST}}/\overline{\text{GRANT LINE}}$ (maximum mode). This line is similar to the $\overline{\text{RQ}}/\overline{\text{GT0}}$ line described above except that the $\overline{\text{RQ}}/\overline{\text{GT0}}$ line has priority when simultaneous requests are received.
HOLD	HOLD LINE (minimum mode). This input, when made high, causes the 8086 to halt at the completion of the current instruction. During the Hold state, the 8086 floats the $\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{INTA}}$, and M/$\overline{\text{IO}}$ command lines, the $\overline{\text{DEN}}$ and DT/$\overline{\text{R}}$ bus control lines, and the multiplexed address/data/status lines to a high impedance state.
HLDA	HOLD ACKNOWLEDGE LINE (minimum mode). This output is made high by the 8086 when a Hold input is acknowledged and the appropriate command, status, bus lines are floated.

Table 3-1. Microprocessor Signals (cont)

SIGNAL NAME	DESCRIPTION
$\overline{WR}$	$\overline{WRITE}$ CONTROL LINE (minimum mode). This output is made low when the 8086 places data on the multiplexed address/data lines (AD0-AD15) during a bus write operation.
M/$\overline{IO}$	MEMORY/$\overline{IO}$ CONTROL LINE (minimum mode). This output is made high by the 8086 when the current bus operation involves memory; this output is made low when the current bus write operation involves an I/O device.
DT/$\overline{R}$	DATA TRANSMIT/$\overline{RECEIVE}$ LINE (minimum mode). This output is made high by the 8086 during a bus write operation; this output is made low during a bus read operation. This line is typically used to control the direction of bus transceivers employed in the system.
$\overline{DEN}$	$\overline{DATA\ ENABLE}$ LINE (minimum mode). This output is made low by the 8086 during the latter portion of a bus read or bus write operation when the multiplexed address/data bus is ready to handle data. This line is typically used to enable bus transceivers employed in the system.
ALE	ADDRESS LATCH ENABLE LINE (minimum mode). This output is made high by the 8086 during T1 of the bus cycle to indicate the presence of address on the time-multiplexed memory bus. The falling edge of ALE is typically used by external devices to latch the address from the data/status information.
$\overline{INTA}$	$\overline{INTERRUPT\ ACKNOWLEDGE}$ LINE (minimum mode). This output is made low by the 8086 during each of two interrupt acknowledge bus cycles which follow an interrupt request on the INTR line. The first $\overline{INTA}$ signal is used by external devices as an indication that the address/data bus is floated; and the second $\overline{INTA}$ signal is used by the external interrupt system to place a byte on the data bus that identifies the source of the interrupt.
INTR	INTERRUPT REQUEST LINE. This input, when made high during the clock period preceding the end of the current instruction, requests external interrupt of the 8086. Interrupt requests at this input may be software masked/enabled.

Table 3-1. Microprocessor Signals (cont)

SIGNAL NAME	DESCRIPTION
NMI	NON-MASKABLE INTERRUPT. This input, when made high for a period of two or more clock cycles, provides a non-maskable interrupt to the 8086.
READY	WAIT STATE CONTROL. This input, when placed at a logic low level (not ready), causes the 8086 to enter a wait state. During the wait state, the 8086 inserts clock pulses to extend the bus cycle as required by the external device selecting the wait state.
$\overline{\text{TEST}}$	WAIT FOR $\overline{\text{TEST}}$. This input allows software to test for some external condition before proceeding, thereby synchronizing operation to some external event. When required by software, this input is checked for a logic low every five clock periods, with the 8086 in an idle state between these checks. When the $\overline{\text{TEST}}$ input is tested and found low, the 8086 continues operation.
RESET	RESET INPUT. When the 8086 receives a positive-going transition at the Reset input, it suspends all operations until the Reset signal goes low, at which time it initializes the system. System initialization includes clearing flags, setting the instruction pointer, setting various internal registers, and emptying the instruction queue.

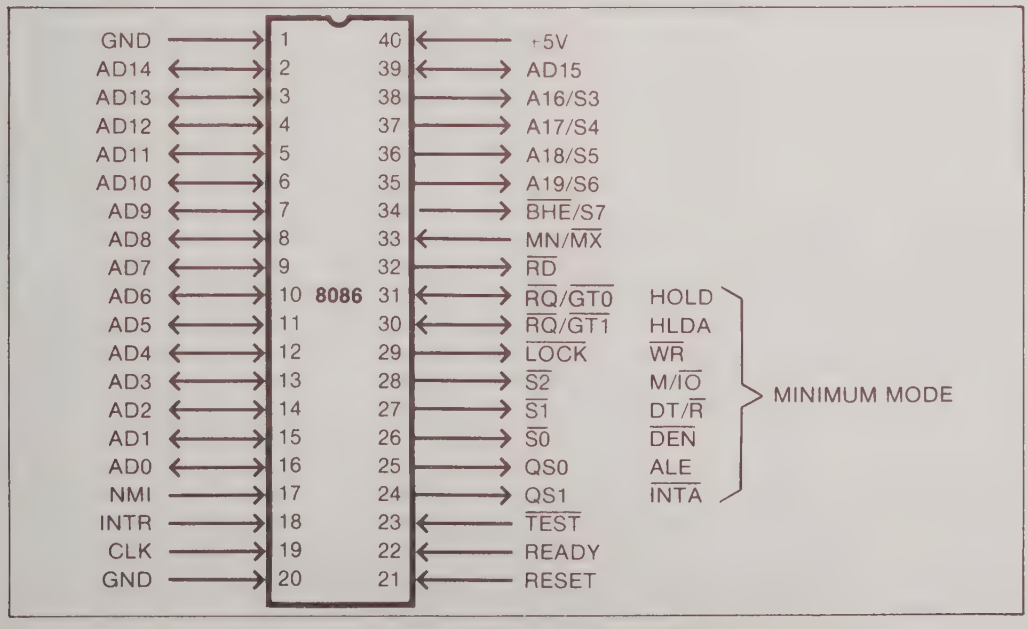


Figure 3-1. 8086 Pin Assignments

Section 4
Operating Information

4-1. INTRODUCTION

This section contains information which pertains to operating the troubleshooter with 8086-based systems. This additional information complements the information in the troubleshooter operator and programming manuals, and covers such items as the following:

- Address space assignment
- Special address functions
- Characteristics of 8086 memory addressing
- Definition and bit assignment of status lines
- Definition of forcing and interrupt lines
- Bit assignment of control lines
- Definition and characteristics of user-writable control lines
- Interrupt handling
- Characteristics of Bus Test, Learn, and Run UUT
- Marginal UUT problems

4-2. GETTING STARTED

After the pod is installed in the UUT and the UUT power is turned on (the troubleshooter power should already be on), it is possible to see the message *POD TIMEOUT - ATTEMPTING RESET*. When this message appears at power on, it is usually caused by faulty status lines (input lines to the pod microprocessor) called pod enable lines. The first thing to try is to reset the UUT. If that does not work, you can disable the enable lines with the troubleshooter Setup function by setting the Setup messages *SET - ENABLE* xxxx? to *NO*.

If you attempt an operation and the message is now *ACTIVE FORCE LINE - LOOP?*, the problem is probably caused by the same faulty status line. You can disable the reporting of this error and continue operation by selecting the troubleshooter Setup function again and setting the Setup message *SET - TRAP ACTIVE FORCE LINE?* to *NO*. For more information about enable lines refer to a later section titled User-Enableable Status Lines. For more information about forcing lines, refer to a later section titled Forcing Lines.

NOTE

Operating the pod with the enable lines disabled may degrade pod performance, especially if the UUT uses Wait states or DMA accesses.

If the message *POD TIMEOUT - ATTEMPTING RESET* remains even after you have disabled the enable lines, the problem may be caused by the UUT power supply (if the UUT power supply is below 3.5V, the pod will disable the inputs to the UUT to protect the UUT) or by a faulty UUT clock. If either of these problems exist, the pod will not operate until they are corrected. If the clock and the power supply are operating properly and the pod still does not work, perform a pod self test as described in Section 2.

If the UUT uses the maximum mode Request/Grant circuitry, it is recommended that you reset the UUT (after power is applied to the troubleshooter and the UUT) and then complete some operation involving the pod, such as Bus Test. This will ensure that the pod and UUT Request/Grant circuitry is synchronized.

4-3. ADDRESS SPACE ASSIGNMENT

4-4. Introduction

The 8086 has 20 physical address lines, AD0 through AD15 and A16/S3 through A19/S6, which allow the 8086 to address one megabyte (1,048,576 bytes) of memory. The 8086 can address either words (16 bits of data) or bytes (8 bits of data). Each physical address is a byte address.

The 8086 pod only accepts even addresses for word accesses; odd addresses default to the next lower (even) address. To access a word at an odd address, the operator must actually use two consecutive byte accesses. During even word accesses, the $\overline{BHE}$ (Bus High Enable) line is low, which enables the high byte at the same time as the low byte.

The address space assignment for the 8086 is listed in Table 4-1. The address space assignments are divided into four categories: word address space, byte address space, I/O word address space, and I/O byte address space.

For most UUTs, the basic word address memory space consists of 00 0000 through 0F FFFE (even words only), and the basic byte address memory space consists of 80 0000 through 8F FFFF. These address spaces access the same memory. The only difference is that operations in the address space 00 0000 through 0F FFFE access a word at a time, and operations in the address space 80 0000 through 8F FFFF access a byte at a time. These address spaces include all the addresses that are needed to test most UUT memory spaces. (Of course, for I/O space you will need the I/O addresses.)

Notice in Table 4-1 that the other word and byte memory address spaces are simply variants on the basic address space that are formed by using the prefixes 1X XXXX, 2X XXXX, 3X XXXX, 9X XXXX, AX XXXX, and BX XXXX (X may be any hexadecimal digit). These variant addresses result from using different "segment" registers to form the physical address, and are not encountered on most UUTs. The 8086 segment registers and the role they play in forming the 8086 addresses is described in a later section titled Role of 8086 Segment Registers in Memory Addressing.

The 8086 can address up to 64K (65,536 decimal) 8-bit ports or 32K (32,768 decimal) 16-bit ports. The I/O space is not segmented; to access a port address, the 8086 places the address on the lower 16 lines of the address bus. As shown in Table 4-1, the troubleshooter may access 16-bit I/O ports at addresses 40 0000 through 40 FFFE (even addresses only, odd addresses default to the next lower even address). The troubleshooter may address 8-bit I/O ports at addresses C0 0000 through C0 FFFF.

Table 4-1. Address Space Assignment

FUNCTION	ADDRESS	DESCRIPTION
WORD ACCESSES*	00 0000 to 0F FFFE	Addresses 16 bits of memory space using the ES (Extra Data Segment) register.
	10 0000 to 1F FFFE	Addresses 16 bits of memory space using the SS (Stack Segment) register.
	20 0000 to 2F FFFE	Addresses 16 bits of memory space using the CS (Code Segment) register.
	30 0000 to 3F FFFE	Addresses 16 bits of memory space using the DS (Data Segment) register.
BYTE ACCESSES**	80 0000 to 8F FFFF	Addresses 8 bits of memory space using the ES (Extra Data Segment) register.
	90 0000 to 9F FFFF	Addresses 8 bits of memory space using the SS (Stack Segment) register.
	A0 0000 to AF FFFF	Addresses 8 bits of memory space using the CS (Code Segment) register.
	B0 0000 to BF FFFF	Addresses 8 bits of memory space using the DS (Data Segment) register.
I/O WORD ACCESSES*	40 0000 to 40 FFFE	Addresses 16 bits of I/O space.
I/O BYTE ACCESSES**	C0 0000 to C0 FFFF	Addresses 8 bits of I/O space.

* Only even addresses are allowed in this address space. Odd addresses default to the next lower address.

** The least significant bit in the address (line AD0) determines which 8-bit byte is read:

AD0 = 0, low byte is read.
AD1 = 1, high byte is read.

4-5. Special Addresses

In addition to the regular address spaces listed in Table 4-1, the pod has some special addresses that are used to access information in the pod or to cause the pod to perform some special functions. These special functions include interrupt handling routines, address specifications for Run UUT, a Quick Looping function, and a Quick RAM Test and Quick ROM Test. The special addresses used by each function are included with the description of each function in later sections.

4-6. Role of 8086 Segment Registers in Memory Addressing

The 8086 has four segment registers that are used to form the effective memory address. The registers are the Extra Data register (ES), Stack register (SS), Code register (CS), and Data register (DS).

The segments are logical units of memory up to 64K bytes long. Each segment is made up of contiguous memory locations and is an independent, separately-addressable unit. Each segment is assigned (by software) a base address which is its starting location in

memory space. Segments may be contiguous to each other, separated from each other, or they may overlap partially or fully.

The 8086 CPU has direct access to four segments at a time. The CS register points to the current code segment from which instructions are fetched. The DS register points to the current data segment, which generally contains program variables. The SS register points to the current stack segment; stack operations are performed on locations in this segment. The ES register points to the current extra data segment, which is typically used for data storage.

During normal 8086 operation, a particular segment register contains the base (or starting) address of a segment. Memory addresses within the segment are created by adding the contents of some other register (which functions as an offset register) or memory location to the base address contained in the segment register.

For example, in the case of the code segment, which typically contains the object code, the base or starting address of the instruction code is held in the code (CS) register while the offset address is maintained in the instruction pointer (similar in function to the program counter in other microprocessors). When the contents of the instruction pointer are added to the contents of the code register, the actual memory address is created for the instruction read from the system bus. Table 4-1 indicates which particular troubleshooter addresses use each segment register.

4-7. Demultiplexing Memory Addresses with the A16/S3 and A17/S4 Lines

The A16/S3 and A17/S4 Lines also play a role in determining the address. The 8086 provides only 20 physical address lines which would ordinarily allow only one megabyte of memory locations to be addressed (F FFFF hexadecimal). However, by demultiplexing the status information contained on lines A16/S3 and A17/S4 in conjunction with using the four segment registers, the available memory locations that may be addressed increases by a factor of four (3F FFFF hexadecimal). Lines A17/S4 and A16/S3 contain address information during the first portion of a bus cycle, and they contain status information during the latter portion of a bus cycle. As shown in Table 3-1, these two lines indicate the segment register used in forming the memory address (presented on the address lines during the first portion of the current bus cycle).

In order to use the status information as the two high-order address bits, the system must first use the particular segment register which yields the bit pattern on lines A17/S4 and A16/S3 required for the desired memory address; then the system must demultiplex these two lines so that A17 and A16 address bits can be separated from the address bits contained in the S4 and S3 status bits.

4-8. STATUS/CONTROL LINES

4-9. Introduction

As part of the task of interfacing the general troubleshooter architecture with the UUT architecture, the pod makes specific assignments between the microprocessor lines and the troubleshooter. These assignments include the following:

- Bit number assignment of status lines
- User-writable control lines
- Bit number assignment of control lines

These assignments, which are summarized on a decal on the bottom of the pod, are listed in Table 4-2 and described in the following paragraphs:

Table 4-2. Status and Control Line Bit Assignments

STATUS LINES			CONTROL LINES		
BIT NUMBER	SIGNAL		BIT NUMBER	SIGNAL	
	MAXIMUM MODE	MINIMUM MODE		MAXIMUM MODE	MINIMUM MODE
11	INT VECT	INT VECT	15	S7	S7
10	**R/G ERROR	—	14	S6	S6
9	$\overline{\text{TEST}}$	$\overline{\text{TEST}}$	13	S5	S5
8	MN/$\overline{\text{MX}}$	MN/$\overline{\text{MX}}$	12	S4	S4
7	PWR FAIL	PWR FAIL	11	S3	S3
6	—	—	10	$\overline{\text{S2}}$	M/$\overline{\text{IO}}$
5	†**HOLD1	—	9	$\overline{\text{S1}}$	DT/$\overline{\text{R}}$
4	**RESET	**RESET	8	$\overline{\text{S0}}$	$\overline{\text{DEN}}$
3	†INTR	†INTR	7	$\overline{\text{BHE}}$	$\overline{\text{BHE}}$
2	NMI	NMI	6	$\overline{\text{RD}}$	$\overline{\text{RD}}$
1	†**HOLD0	†**HOLD	5	*SPECIAL	—
0	†**READY	†**READY	4	*$\overline{\text{GT1}}$	*HLDA
			3	*$\overline{\text{GT0}}$	—
			2	*QS1	*$\overline{\text{INTA}}$
			1	*QS0	ALE
			0	*$\overline{\text{LOCK}}$	$\overline{\text{WR}}$

†USER ENABLEABLE
*USER WRITABLE
**FORCING LINES

NOTE

The terms status and control as used in this manual differ in meaning from the same terms used in the microprocessor manufacturer's literature. The troubleshooter considers input lines to the microprocessor to be status lines, and output lines to be control lines.

4-10. Status Line Bit Assignments

When a Read Status operation is performed, the troubleshooter displays the result in binary form, where "1" indicates a logic high status line and a "0" indicates a logic low status line. To determine which characters of the display correspond to specific status lines, refer to Table 4-2.

NOTE

The status line assignments vary depending on the operating mode of the 8086 (maximum or minimum mode). Assignments for each mode are listed in Table 4-2 and on the pod decal.

For example, if a Read Status operation is performed and the Ready (Bit 0) and Interrupt Vector (Bit 11) lines are low with all other status lines high, the troubleshooter displays the message READ @ STS = 0111 1011 1110 OK. Bit numbers 0 (Ready) and 11 (Interrupt Vector) are zero to indicate a logic low, while other meaningful bits are 1 to

indicate a logic high. Bit 6, which has no meaning as an 8086 status line, is always represented by a zero in the troubleshooter display message.

NOTE

The Power Fail, R/G Error, Interrupt Vector, HOLD0, and HOLD1 bits do not represent particular 8086 signals, but are generated within the pod. (Refer to the section titled Status Lines Unique to the Pod.) Bits 1 and 5 (HOLD0 and HOLD1) indicate the latched condition of these lines when operating in the maximum mode.

4-11. User-Enableable Lines

The 8086 has several inputs which the operator can individually enable or disable (Ready, Hold, and INTR in the minimum mode, and Ready, $\overline{RQ}/\overline{GT0}$, $\overline{RQ}/\overline{GT1}$, and INTR in the maximum mode) by selecting the proper Setup message using the Setup function of the troubleshooter.

NOTE

The two maximum mode enable lines, $\overline{RQ}/\overline{GT0}$ and $\overline{RQ}/\overline{GT1}$, are related to the status lines HOLD0 and HOLD1. HOLD0 and HOLD1 are latches within the pod that indicate when the $\overline{RQ}/\overline{GT0}$ and $\overline{RQ}/\overline{GT1}$ lines have been pulsed. Because the request pulse is usually very brief, the condition of the latches HOLD0 and HOLD1 is reported in the Read Status operation. $\overline{RQ}/\overline{GT0}$ and $\overline{RQ}/\overline{GT1}$ are the physical lines that are enabled, and are the names used in the Setup messages SET-ENABLE xxxxxx?

When these inputs are disabled, UUT-generated signals appearing at these inputs are prevented from affecting troubleshooter and pod operation. For example, a stuck-low Ready line would cause the 8086 within the pod to remain in a wait state, preventing normal troubleshooter/pod operation. When the Setup function of the troubleshooter is used to disable this input to the 8086 (by means of hardware in the pod), the Ready signal is prevented from reaching the 8086 within the pod, thus allowing the troubleshooter/pod interactions to take place normally.

Any of these status lines may be enabled or disabled using the troubleshooter Setup function as described in the troubleshooter operator manual. The relevant Setup display message is SET-ENABLE xxxxxx? where xxxxxx is the name of the status line (there is one Setup message for each enable line). Pressing the YES key on the troubleshooter enables the status line; pressing the NO key disables the status line.

NOTE

During troubleshooter Setup, selecting the message SET-ENABLE xxxxxx? NO prevents the enable line from affecting the operation of the pod (although the pod can still detect whether the line is high or low). This differs from selecting the troubleshooter Setup message SET-TRAP ACTIVE FORCE LINE? NO which does not prevent an enable line from affecting the operation of the microprocessor, but does prevent the active condition from being reported on the troubleshooter display.

4-12. Status Lines Unique to the Pod

Several of the status lines reported during a Read Status operation are generated by the pod and not by the UUT. These lines consist of Power Fail, R/G Error, Interrupt Vector, HOLD0, and HOLD1. Since these lines are a function of the pod and not the 8086, they are not described in Section 3, but are described in the following paragraphs:

4-13. POWER FAIL STATUS LINE

The Power Fail status line is set high by the pod whenever UUT power supply voltages drops below 4.5V or rises above 5.5V.

4-14. R/G ERROR STATUS LINE (MAXIMUM MODE ONLY)

The Request Grant Error (R G ERROR) line is set high whenever the pod determines that the pod and UUT Request-Grant signals are not synchronized. This condition can be caused if the pod is reset and the UUT is not, or by a Write Control operation which toggles the $\overline{GT0}$ or $\overline{GT1}$ line. A reset from the UUT to the pod clears this condition.

4-15. INTERRUPT VECTOR STATUS LINE

The Interrupt Vector line is set high by the pod whenever a new unread interrupt vector is available. This vector may be read as described in a part of Section 4 titled Using Special Addresses to Read Interrupt Type and Cascade Address.

4-16. HOLD0 AND HOLD1 STATUS LINES (MAXIMUM MODE ONLY)

The HOLD0 or HOLD1 line is set high by the pod to signify that the $\overline{RQ}$ $\overline{GT0}$ or $\overline{RQ}$ $\overline{GT1}$ line has been pulsed, which indicates that a Hold operation has been requested. The HOLD0 or HOLD1 line is cleared at the end of the Request Grant release operation, or by a UUT reset.

4-17. Forcing Lines

One of the troubleshooter error messages that may be displayed is the message *ACTIVE FORCE LINE* (@ aaaa)-*LOOP?* Forcing lines are a special category of status lines which, when active, can force the microprocessor into some specific state or action which causes a pod timeout. The *ACTIVE FORCE LINE* error message helps isolate status lines which are not functioning properly.

The forcing lines consist of Ready, HOLD0, HOLD1, Reset, and R G Error in the maximum mode, and Ready, Hold, and Reset in the minimum mode. Notice that the Ready, Hold, HOLD0, and HOLD1 lines are user-enableable lines. If these user-enableable lines are disabled (using the troubleshooter Setup function), their inputs to the pod microprocessor are disabled, but the pod continues to monitor their condition; if they are active (high), the pod reports to the troubleshooter that a forcing line is active. Note that if these user-enableable lines are enabled (using the troubleshooter Setup function), they are not considered forcing lines even when they are active.

The Reset line is hardware-disabled except during Run UUT, but it is monitored and reported as an active forcing line if it is active during non-Run UUT operation.

A typical way the *ACTIVE FORCE LINE* message would be encountered is as follows. Assume the pod and troubleshooter are connected to a UUT and the operator first turns on the power to the troubleshooter and the UUT (at power on the user-enableable lines are all enabled). When attempting a Read operation with the troubleshooter, the error message *POD TIMEOUT - ATTEMPTING RESET* appears. The operator enters the troubleshooter Setup function, disables the Ready enableable line, and again attempts a Read operation. This time the troubleshooter displays the message *ACTIVE FORCE LINE* @ aaaa-*LOOP?* (aaaa is the address specified for the Read operation).

To find out which line or lines are the active forcing lines, the operator presses the MORE key and the message *STS BTS 0000 0000 0001* is displayed. According to Table 4-2, the Ready line is the 0 bit, so the display message indicates that the Ready line is the active forcing line that is not functioning properly. While the Ready line was enabled, its improper function caused the pod timeout to occur. After the Ready line was disabled, its improper function caused the *ACTIVE FORCE LINE* error message to appear.

NOTE

It is possible to disable the reporting of active forcing lines by selecting the troubleshooter Setup function message SET-TRAP ACTIVE FORCE LINE? NO. The pod will still monitor the lines, but the troubleshooter will not interrup its operation to display the ACTIVE FORCE LINE error message on the display. Sometimes it is useful to disable the reporting of active forcing lines, particularly if the information is not needed by the operator. Reporting of the error message is enabled at power on.

4-18. Interrupt Lines

Interrupt lines for the 8086 consist of INTR and NMI. The INTR input may be enabled by the operator as described in the previous section titled User-Enableable Status Lines. The NMI input is hardware-disabled except during operation in the Run UUT mode. Although disabled, the NMI input is routinely checked by the pod software and reported to the troubleshooter if held high by the UUT.

NOTE

Reporting of the active interrupt error message is disabled at power on. It is possible to enable the reporting of active interrupt lines by selecting the troubleshooter Setup function message SET-TRAP ACTIVE INTERRUPT? YES. Sometimes it is useful to enable the reporting of active interrupts if the information is needed by the operator.

4-19. User-Writable Control Lines

The 8086 has several control lines which the troubleshooter can set high or low with the Write Control function. This feature is used by Bus Test to check lines which cannot be toggled by normal read and write operations. It is also useful for helping troubleshoot these lines. The Write Control function is described in the following paragraphs as it pertains to the 8086 pod. Note that the Write Control function only sets a line high or low for approximately one UUT access, just long enough to verify that it can be driven.

4-20. CONTROL LINES IN MAXIMUM AND MINIMUM MODE

When the UUT operates with the 8086 in the maximum mode, the user-writable control lines consist of $\overline{\text{Lock}}$, QSO, QS1, $\overline{\text{GT0}}$, $\overline{\text{GT1}}$, and Special. When the 8086 operates in the minimum mode, the user-writable control lines consist of $\overline{\text{INTA}}$ and HLDA. Refer to Table 4-2 for a summary of the signal names and bit numbers.

4-21. $\overline{\text{GT0}}$, $\overline{\text{GT1}}$, AND SPECIAL CONTROL LINES (MAXIMUM MODE ONLY)

During normal 8086 operation in the maximum mode, the GT0 and GT1 lines become control lines only after first acting as status lines and receiving a pulse input signifying a request to grant bus usage to some other device in the system. Since these lines do not always function as control lines, they are not written to during a Bus Test by the troubleshooter.

The $\overline{\text{GT0}}$ and $\overline{\text{GT1}}$ lines may be written to by means of the Write Control function only if the bit corresponding to the Special line (Bit 5) is specified as 1. Note that this is the only purpose for the Special line, to allow the $\overline{\text{GT0}}$ and $\overline{\text{GT1}}$ lines to be written by the Write Control function. This feature helps prevent $\overline{\text{GT0}}$ or $\overline{\text{GT1}}$ from being inadvertently written low, which could cause unpredictable behavior in the UUT. After the $\overline{\text{GT0}}$ or $\overline{\text{GT1}}$ lines are written low with the Write Control function, it is a good idea to reset the UUT to ensure synchronous operation.

NOTE

If the Write Control function is selected and the $\overline{GT0}$ and $\overline{GT1}$ line bit numbers (Bits 3 and 4) are set to 0 in the specification, it is the pod that checks to see if the Special line bit number (Bit 5) is set to 1, not the troubleshooter. Therefore, it is possible to key in a Write Control specification which appears to write $\overline{GT0}$ or $\overline{GT1}$ low (such as 0000 0000), but unless Bit 5 is specified as 1 (0010 0000), the pod will not actually write the lines low.

4-22. Control Line Bit Assignments

There are two troubleshooting functions which require the entry of binary digits to identify user-writable control lines. These functions are Write Control and Data Toggle Control.

When performing or programming either of these two functions, the operator is prompted for a binary number to identify the control line(s) to be written. Table 4-2 shows that these lines are assigned to bit numbers 0 through 5 when the 8086 is in the maximum mode, and bit numbers 2 and 4 when the 8086 is in the minimum mode.

For example, to perform a Write Control operation which writes all six maximum mode control lines high, the operator enters the string 111111 when the troubleshooter prompts with the message *WRITE @ CTL =*. To write any of the lines to the low state, the operator enters a 0 in place of 1 at the bit position which corresponds to the particular control line (the $\overline{GT0}$ and $\overline{GT1}$ lines require that Bit 5 be specified as 1; refer to the previous section titled $\overline{GT0}$, $\overline{GT1}$, and Special Control Lines for more information). The control line bit assignments are listed in Table 4-2.

To perform a Write Control operation which writes both minimum mode lines high, the operator enters 1X1XX (X indicates either a 1 or a 0). To write both lines low the operator enters 0X0XX.

When performing Bus Test or various other troubleshooter functions, the troubleshooter may detect that one or more control lines are not drivable. For example, suppose that during a Bus Test the troubleshooter detects that the 8086 $\overline{Lock}$ line (maximum mode) is not drivable. The troubleshooter displays the message *CTL ERR 0000000 00000001-LOOP?* The zeros and ones correspond to the bit numbers assigned to the control lines as listed in Table 4-2. The first bit, Bit 0, is set to 1 because the $\overline{Lock}$ line was detected as not drivable. The other error messages that pertain to non-drivable control lines use the same bit number assignments as listed in Table 4-2.

4-23. SPECIAL FUNCTIONS UNIQUE TO THE 8086 POD

4-24. Introduction

There are two tests and one troubleshooting function that are unique to the 8086 pod: the Quick Looping function, Quick RAM Test, and Quick ROM Test. Unlike the ordinary Looping function, RAM Test, and ROM Test, the software routines that control the Quick functions reside in the pod and not in the troubleshooter. The operator selects these functions by writing to the special addresses listed in Table 4-3.

As their names imply, the advantages of the Quick functions is that they are executed more quickly than the corresponding ordinary functions. It should be noted, however, that the diagnostics performed by the pod during the execution of the Quick functions are less rigorous than the diagnostics performed during the execution of the ordinary functions. Each function is described in the following paragraphs.

Table 4-3. Special Functions Unique to the 8086 Pod

FUNCTION	SPECIAL ADDRESSES AND OPERATIONS	DESCRIPTION OF USE
QUICK LOOPING FUNCTION	READ @ 1XX XXXX WRITE @ 1XX XXXX = YYYY	A read or write at address 1XX XXXX will cause the pod to perform a quick looping read or write (with data YYYY) at address XX XXXX. UUT system errors are reported only during the first execution of read or write, and not during the succeeding execution loops.
QUICK ROM TEST	Specifying the Test	
	WRITE @ 3XX XXXX = 0	Specifies starting address (XX XXXX) for Quick ROM Test.
	WRITE @ 3YY YYYY = Z1	Specifies ending address (YY YYYY) and the address increment (Z) for Quick ROM Test. If Z = 0, the address increment default to 2.
	Requesting Information About Test Execution	
	READ @ ENTER	After test has been specified, operator can request information about execution results by pressing the keys READ ENTER (address specification is defaulted). The resulting code that is displayed on the troubleshooter indicates the following: CODE MEANING 0000 No test requested 00A0 Aborted test, new command entered 00A1 Aborted test, illegal data in command 00A2 Aborted test, illegal address in command 00B0 Busy, test in progress 00C0 Complete, no errors 00C1 Complete, inactive bits detected
	READ @ 300 0000 READ @ 300 0002	Low word of starting address High word of starting address
	READ @ 300 0004 READ @ 300 0006	Low word of ending address High word of ending address
	READ @ 300 000C READ @ 300 000E	Checksum* Hex mask indicating inactive bits detected during test
	READ @ 300 0010	Most recent code returned (same as code obtained by READ @ ENTER)

*The checksum obtained by the Quick ROM Test is not related to the ROM signature obtained by the ordinary ROM Test.

Table 4-3. Special Functions Unique to the 8086 Pod (cont)

FUNCTION	SPECIAL ADDRESSES AND OPERATIONS	DESCRIPTION OF USE
QUICK RAM TEST	Specifying the Test	
	WRITE @ 2XX XXXX = 0	Specifies starting address (XX XXXX) for Quick RAM Test.
	WRITE @ 2YY YYYY = Z1	Specifies ending address (YY YYYY) and the address increment (Z) for Quick RAM Test. If Z = 0, the address increment defaults to 2
	Requesting Information About Test Execution	
	READ @ ENTER	After test has been specified, operator can request information about execution results by pressing the keys READ ENTER (address specification is defaulted). The resulting code that is displayed on the troubleshooter indicates the following CODE MEANING 0000 No test requested 00A0 Aborted test, new command entered 00A1 Aborted test, illegal data in command 00A2 Aborted test, illegal address in command 00B0 Busy, performing read/write check 00B1 Busy, performing address decoding check 00C0 Complete, no errors 00F0 Failed, read/write error 00F1 Failed, address decoding error
	READ @ 200 0000 READ @ 200 0002	Low word of starting address High word of starting address
	READ @ 200 0004 READ @ 200 0006	Low word of ending address High word of ending address
	READ @ 200 0008 READ @ 200 000A	Low word of error address High word of error address
	READ @ 200 000C READ @ 200 000E	Data expected at error address Actual data returned from error address
	READ @ 200 0010	Most recent code returned (same as code obtained by READ @ ENTER)
	READ @ 200 0012	Hex mask where ones correspond to bad data bits from read/write failure or address decoding failure*

*If a read/write failure occurred (code F0), the hex mask indicates bits that are not read/writeable both high and low.

*If an address decoding failure occurred (code F1), the hex mask corresponds to address bits that are probably at fault. For example, a hex mask of 0080 indicates that address bit 7 is at fault. Note that the first four bits in the hex mask correspond to address bits 16 through 19 as well as to address bits 0 through 3.

4-25. Quick Looping Read or Write

The Quick Looping read or write function is useful for enhanced viewing on an oscilloscope that is synchronized to the TRIGGER OUTPUT pulse (available on the troubleshooter rear panel). If a signal trace on the oscilloscope screen is dim due to a low repetition rate, the Quick Looping function can increase the repetition rate to make the signal trace much more visible.

To select the Quick Looping function, specify a read or write operation at the address 1XX XXXX. The pod first performs a read or write operation at address XX XXXX in the normal manner, reporting to the troubleshooter any UUT system errors detected (such as *ACTIVE FORCE LINE*, or *CTL ERR*, etc.); then the pod enters the Quick Looping mode where the read or write operation is performed several times faster than the ordinary Looping function specified by pressing the LOOP key on the troubleshooter keyboard. During the Quick Loop, the pod does not check for any UUT system errors that may occur. The Quick Loop continues until the operator selects another operation.

For example, if the operator specifies the operation *READ @ 12F F000*, the pod will perform a looping read operation at the address 2F F000. If the operator specifies the operation *WRITE @ 180 B007 = 2F*, the pod will perform a looping write operation at address 80 B007, writing the data 002F.

The Quick Looping function may be used with read or write operations at any of the valid 8086 addresses listed in Table 4-1, and any of the special interrupt function addresses listed in a later section titled Interrupt Handling. The Quick Looping function may not be used with any of the other troubleshooting functions or tests.

If both error reporting and the Quick Looping feature are desired, you may apply the ordinary troubleshooter Looping function to the Quick Looping read or write, such as *READ @ 1XX XXXX LOOP*. The troubleshooter will command read operations at address XX XXXX at the normal looping speed with full error reporting. For every ordinary read operation, the pod will interject a few Quick Looping read operations (with no error reporting) which will enhance oscilloscope viewing.

4-26. Quick RAM Test

The Quick RAM Test allows the operator to test RAM address blocks more quickly than with the tests RAM Short or RAM Long. The Quick RAM Test is considerably faster than the RAM Short test and is almost as rigorous. The Quick RAM test is particularly well suited for programming applications.

The Quick RAM Test consists of two phases; the first test phase is a read-write check, and the second phase is an address decoding check. The read-write check is performed by writing and reading a one and a zero from each bit of each test addresss to ensure that there are no bits held high or low. After the read-write check is completed, a unique word remains at each address. For the address decoding check, the pod reads each address and compares the data read with the unique word that is expected.

The starting and ending addresses for the Quick RAM Test are specified in a slightly different manner than for the usual RAM Test. The starting and ending addresses are specified by writing to the special addresses listed in Table 4-3. To specify the starting address, enter *WRITE @ 2XX XXXX = 0* where XX XXXX is the address. To specify the ending address, enter *WRITE @ 2YY YYYY = Z1*, where YY YYYY is the address and Z is the desired address increment value (in bytes). If Z is not specified, the default value is two bytes. Either word addresses or byte addresses may be specified. The ending address must be greater than the starting address, and both addresses must be even for word

addresses. Both addresses must use the same segment register. The address increment value must be even for word addresses.

The Quick RAM Test begins execution as soon as the operator completes the entry of the ending address. During or after the execution of the test, the troubleshooter will not display any information about the test progress or test results unless requested by the operator. The test may be aborted before completion by selecting another operation.

To determine if the Quick RAM Test is still in progress, or has been completed, aborted, or failed, specify *READ @ ENTER* (which defaults the address specification). The troubleshooter displays a code indicating the status of the test or the test results. A list and description of the codes returned by *READ @ ENTER* is provided in Table 4-3.

For more information about the test results, the operator may specify read operations at the special addresses listed in Table 4-3. It is a good practice to specify the *READ @ ENTER* first to find out if the test has been completed before reading at any of the special addresses. Unless the test has been completed (or failed), the information contained at the special addresses will pertain to a previous test rather than the current test.

The following example shows how to specify the Quick RAM Test over the address block 5000 through 5FFE, with an address increment of 4 bytes:

WRITE @ 200 5000 = 0

WRITE @ 200 5FFE = 41

Note that the Quick RAM Test has more flexibility than the ordinary RAM Test with regard to the address increment and specification. For example, only even word addresses may be tested with the ordinary RAM Test since the address specification is restricted to even addresses. However, consider the following Quick RAM Test specification:

WRITE @ 280 F001 = 0

WRITE @ 280 F4FF = 1

The two preceding write operations specify a Quick RAM Test which is to take place at the odd addresses over the address block 80 F001 through 80 F4FF. The value of the address increment is the default value, two bytes.

NOTE

When performing the Quick RAM Test on blocks of RAM that are less than 32K bytes in size, specify byte addresses with an increment of one, such as WRITE @ 28X XXXX = 11. The reason for testing each byte is that it more thoroughly exercises the high data bits.

4-27. Quick ROM Test

The Quick ROM Test allows the operator to test ROM address blocks more quickly than with the ordinary ROM Test. When the Quick ROM Test is performed, the pod obtains a checksum that may be compared with a checksum obtained by performing the Quick ROM Test over the same address block in a known good UUT. Note that this checksum is not the same value as the signature that is obtained with the ordinary ROM Test.

The Quick ROM Test does not perform as rigorous and thorough manipulation of the ROM as does the ordinary ROM Test, nor does the Quick ROM Test have as extensive error reporting. However, the Quick ROM Test can detect inactive data bits, and the checksum can be used to detect a faulty ROM device with a fairly high degree of confidence.

The Quick ROM Test is specified in a manner similar to the Quick RAM Test. To specify the starting address, enter *WRITE @ 3XX XXXX = 0* where XX XXXX is the address. To specify the ending address, enter *WRITE @ 3YY YYYY = Z1* where YY YYYY is the address and Z is the desired address increment value (in bytes). If Z is not specified, the default address increment is two bytes. Either word addresses or byte addresses may be specified. The ending address must be greater than the starting address, and both addresses must be even for word addresses. Both addresses must use the same segment register. The address increment must be even for word addresses.

Like the Quick RAM Test, the Quick ROM Test may be aborted by selecting another operation. To determine the status or results of the Quick ROM Test, specify *READ @ ENTER*. The troubleshooter will display one of the codes listed in Table 4-3. To obtain more information about the results of the test, read operations may be performed at the special addresses listed in Table 4-3.

4-28. RUN UUT MODE

The Run UUT mode allows the pod to emulate the UUT microprocessor by executing a program directly from UUT memory. When the operator selects Run UUT, the operator may either explicitly specify the address where execution begins, or the operator may invoke the Run UUT default execution address which is supplied by the pod. Each of these cases are described in the following paragraphs. Note that interrupts are always disabled when the troubleshooter enters the Run UUT mode. The special addresses that pertain to the Run UUT mode are listed in Table 4-4.

4-29. Run UUT Default Execution Address

If the Run UUT mode is selected and the operator does not specify the execution address, the pod supplies the address 000F FFF0 as the execution address. Execution at this address sets the segments registers to their reset values: 0000 for the ES, SS, and DS registers, and FFFF for the CS register; the offset address is set to 0000.

Note that the operator may change the Run UUT default address with the troubleshooter Setup function by entering the desired address for the Setup message *SET-RUN UUT @ FFFF0-CHANGE?*

4-30. Specifying Execution Address

The operator may select the address where Run UUT execution begins by entering an address when Run UUT is first selected. All addresses that are executed in Run UUT are formed using the CS register, so the segment prefix does not matter if an even address is specified. (The segment prefix is the sixth hexadecimal digit, such as 2X XXXX for the CS register, or 3X XXXX for the DS register.) If an odd address is specified, a segment prefix specifying a byte access should be used (such as 8X XXXX, 9X XXXX, etc.).

Whenever any address other than the default address (F FFF0) is specified, the pod considers the offset address to be equal to the four least significant digits, and the CS register contents to equal the fifth hexadecimal digit followed by three zeros. Consider the following examples:

Table 4-4. Special Addresses for Run UUT Mode

FUNCTION	ADDRESS	DESCRIPTION
DEFAULT EXECUTION ADDRESS	F FFF0	This is the Run UUT default address that the pod supplies if the operator does not specify an execution address. Run UUT execution at F FFF0 sets all segment registers to their reset values (0 for ES, SS, and DS registers; FFFF for CS register) and sets the offset to 0.
SPECIFYING EXECUTION ADDRESS	XYZZZZ	Run UUT execution addresses are always formed using the CS register. If execution is specified at address XYZZZZ (assuming it does not equal F FFF0), the offset value is ZZZZ and the CS value is Y000 (the value of X should be 0 for an even address, and 8 for a byte address).
SPECIFYING EXECUTION ADDRESS AND SEGMENT REGISTER CONTENTS	F0 XXXX F0 0000 F0 0002 F0 0004 F0 0006	The contents of any or all of the segment registers may be specified by writing the desired values to the following addresses before executing Run UUT: F0 0000 = ES register contents F0 0002 = SS register contents F0 0004 = CS register contents F0 0006 = DS register contents After the desired values are written to the above addresses, execute Run UUT at the address F0 XXXX, where XXXX is the desired offset address. The specified contents are loaded into the segment registers. As usual, the Run UUT execution addresses are formed using the CS register: CS register contents are shifted left four bits, then added to the offset address to form the execution address.

ADDRESS ENTERED BY OPERATOR	CS REGISTER CONTENTS	OFFSET
000A BCDE	A000	BCDE
003A BCDE	A000	BCDE
000F FFFF	F000	FFFE
008F FFFF	F000	FFFF
000F FFF0 (default address)	FFFF	0000

Compare the third and fourth addresses in the preceding list. In the third address an odd word address is specified *(000F FFFF)*. However, only even words are allowed so the troubleshooter forces the address to the next lower even address (which causes the offset to be even, FFFE). In the fourth address an odd byte address is specified *(008F FFFF);* odd bytes are allowed, so the offset remains odd as specified, FFFF.

Notice also in the preceding list that if the operator enters the default address (F FFF0), the CS register contents are set to the reset value and the offset is 0000.

4-31. Specifying Segment Register Contents Before Execution

It is possible to specify the contents of any or all of the segment registers before the Run UUT execution begins. To specify the segment register contents, write the desired 16-bit value to the following special addresses before selecting Run UUT (these special addresses are also listed in Table 4-4):

F0 0000 = ES register initial contents

F0 0002 = SS register initial contents

F0 0004 = CS register initial contents

F0 0006 = DS register initial contents

Read operations may be performed at these special addresses to confirm that they contain the desired values. After the desired values are written to the above addresses, specify Run UUT at the address F0 XXXX (XXXX equals the offset address). When Run UUT begins, the initial contents of the segment registers will equal the values at the special addresses.

Note that during execution of Run UUT, no information is passed back to these special addresses; even though the segment register contents change, the values at the special addresses are unchanged by Run UUT. As usual, the Run UUT execution addresses are formed using the CS register contents and the offset address. (The value in the CS register is shifted left four bits and then added to the offset address.)

4-32. DEFAULT ADDRESSES FOR LEARN AND BUS TEST

Most troubleshooter operations require operator entry of address information. If the information is not specified, the troubleshooter supplies default address information. The following paragraphs describe default addresses that are unique to the pod for the Learn operation and the Bus Test. The default execution address for the Run UUT mode is described in a previous section titled Run UUT Mode. Other default addresses not mentioned in this manual are described in the troubleshooter operator manual and apply to all pods.

4-33. Learn Operation Default Address

If the Learn operation is selected and the operator does not specify the starting and ending addresses for the operation, the pod specifies the default address spaces of 0000 0000 through 000F FFFE for the basic word address space and 0040 0000 through 0040 FFFE for the I/O word space. The Learn operation is performed over these address spaces.

4-34. Default Address for Data Line Testing for Bus Test

When the operator selects Bus Test, there is no address that is explicitly required. However, as part of Bus Test, the data lines are tested at a particular address supplied by the pod. For the 8086 pod, the data line testing occurs at address 0000. Note that the operator may change this address with the troubleshooter Setup function by entering the desired address for the Setup message *SET-BUS TEST @ 0000-CHANGE?*

4-35. INTERRUPT HANDLING

4-36. Introduction

During normal operation, the 8086 acknowledges an interrupt request by executing two consecutive interrupt acknowledge bus cycles. The first cycle notifies the interrupting device, typically a programmable interrupt controller (PIC), that the request has been

honored. During the second cycle, the interrupting device responds by placing an eight-bit byte of data on the bus which contains the interrupt type (0-255) associated with the device requesting service. The 8086 reads this interrupt type and uses it to call the corresponding interrupt procedure.

NOTE

The interrupt type assignment is made when the PIC is initialized by software in the 8086.

In order to link different types of interrupts (required by different types of external devices) to corresponding interrupt routines held in memory, the 8086 reserves the first 1K bytes of low memory for an interrupt pointer (vector) table. This table may contain up to 256 entries, one for each interrupt type that can occur in the system. Each entry in the table is accessed by the corresponding interrupt type code, and is a double-word pointer containing the memory address of the procedure required to service that type of interrupt. One word of the pointer contains the base address of the segment containing the procedure. The other word contains the offset of the procedure from the base address (described in a previous section titled Role of 8086 Segment Registers in Memory Addressing).

4-37. Using Special Addresses to Read Interrupt Type and Cascade Address

Using the Setup troubleshooter function, the operator has the option of enabling or not enabling interrupts. When interrupts are enabled, the pod responds to an interrupt request from the UUT by means of an interrupt acknowledge cycle, although no attempt is made to execute the interrupt service routine of the UUT. The pod does, however, record the interrupt type placed on the bus by the UUT.

The eight-bit interrupt type can be shown on the troubleshooter display by performing a read operation at the special addresses 50 0000 or 60 0000. (The special addresses that pertain to interrupt handling are listed in Table 4-5.) A read at address 50 0000 also reenables interrupts, and it clears the INT VECT status line low. (The status lines may be read with the Read Status operation on the troubleshooter; INT VECT is Bit 11 on the display message.) A read at address 60 0000 does not reenable interrupts, nor does it set the INT VECT status line low.

The 16-bit cascade address can be shown on the troubleshooter display by performing a read at address 50 0002 or 60 0002. The cascade address is supplied by the interrupt controller (PIC) during the interrupt acknowledge routine to indicate the interrupting device.

Note that whenever an interrupt is acknowledged, succeeding interrupts are temporarily disabled until the current interrupt type is read using the special address 50 0000. This prevents multiple interrupts from removing the current interrupt type before it can be read by the troubleshooter.

4-38. Forcing the Execution of the Interrupt Acknowledge Routine

The pod can be forced to execute an interrupt acknowledge routine at any time by performing a write operation (with any data) at any of the following addresses: 50 0000, 50 0002, 60 0000, or 60 0002. Interrupt acknowledge occurs regardless of whether the UUT requested an interrupt, and regardless of whether the interrupts were enabled with the troubleshooter Setup function. The interrupt information returned to the 8086 can be read as described in the previous section titled Using Special Addresses to Read Interrupt Type and Cascade Address.

Table 4-5. Special Addresses for Interrupt Handling

FUNCTION	ADDRESS	DESCRIPTION
CAUSING INTERRUPTS	50 0000 50 0002 60 0000 60 0002	A write (with any data) to any of these addresses causes the pod to execute an Interrupt Acknowledge routine. The interrupt information obtained by the execution of this routine may be read at addresses 50 0000 or 60 0000.
READING INTERRUPT TYPE	50 0000	A read at this address causes the troubleshooter to display the last interrupt type received. The read will also reenable the interrupt after it is read, and clear the Interrupt Vector status bit (INT VECT is Bit 11).
	60 0000	A read at this address causes the troubleshooter to display the interrupt type, but it will not reenable the interrupt nor clear the Interrupt Vector status bit.
READING CASCADE ADDRESS	50 0002 60 0002	A read at either of these addresses causes the troubleshooter to display the cascade address that was on the address bus during execution of the last Interrupt Acknowledge routine.

4-39. PROBE AND SCOPE SYNCHRONIZATION MODES

The operator may use the troubleshooter Synchronization function (selected with the SYNC key) to synchronize probe operation or the rear panel TRIGGER OUTPUT pulse to events on the pod microprocessor bus. There are three synchronization modes available:

A = Address Sync

D = Data Sync

F = Free-Run

If address sync is selected, both the probe and scope trigger output on the troubleshooter are synchronized to the address portion of the UUT access. The scope trigger will pulse low shortly before the UUT access begins, and pulse high at the end of the address portion of the cycle. If the probe stimulus mode is selected, the probe will pulse at the selected level (high or low) for the time between the two scope trigger pulses described above. For probe response, the probe will latch on the signal level present at its tip at the time of the second or high-going scope trigger pulse.

If data sync is selected, the scope trigger will pulse low at the start of the data portion of the UUT access (the end of the address portion). It will pulse high at the end of the data cycle (the trailing edge of the Data Enable pulse). Probe stimulus and response behave in a manner similar to that described for address sync.

One recommended method for using the scope synchronization is to synchronize on the negative edge of the scope trigger using the address sync mode. This will trigger the scope at a time slightly before the UUT access, allowing you to see the entire UUT access. If you cannot verify that the scope is synchronized to the correct edge, use the pulse stimulus mode of the probe and display the signal on the scope. Since the probe stimulus pulses only occur during the UUT access (when the address sync mode is selected), you can easily check whether the scope is synchronized to the correct edge. After verifying the correct

synchronization, use the scope to look at the Address Latch pulse or the Data Enable pulse to verify when the signal of interest should be valid. The relationship of these signals and the UUT access is described and illustrated in Section 5.

If the signal image on the scope is dim because of a low repetition rate, use the Quick Looping function described in the previous section titled Quick Looping Read or Write. The Quick Looping function will increase the repetition rate considerably, which makes the signal much easier to see on the scope.

Note that the scope trigger output pulses are always synchronized to either address sync or data sync, even if free-run is selected. If free-run is selected, the scope trigger output pulses remain synchronized to the previous sync mode selected (either address sync or data sync). At power on the probe is in free-run, but the scope trigger output pulses are synchronized to data sync.

If free-run is selected, the probe stimulus pulses are generated at a frequency of approximately 1 kHz with a 1% duty cycle.

4-40. PROBLEMS DUE TO A MARGINAL UUT

4-41. Introduction

The pod is designed to approximate, as closely as possible, the actual characteristics of the microprocessor it replaces in the UUT. However, the pod does differ in some respects. In general, these differences tend to make marginal UUT problems more visible. A UUT may operate marginally with the UUT microprocessor installed, but exhibit errors with the pod plugged in. Since the pod differences tend to make marginal UUT problems more obvious, the UUT is easier to troubleshoot. Various UUT and pod operating conditions that may reveal marginal problems are described in the paragrahs which follow.

4-42. UUT Operating Speed and Memory Access

Some UUTs operate at speeds which approach the time limits for memory access. The pod contributes a slight time delay which causes memory access problems to become apparent.

4-43. UUT Noise Levels

As long as the UUT noise level is low enough, normal operation is unaffected. Removing the UUT from its chassis or case may disturb the integrity of the shielding to the point where intolerable noise could exist. The pod may introduce additional noise. In general, marginal noise problems will actually be made worse (and easier to troubleshoot) through use of the pod and troubleshooter.

4-44. Bus Loading

The pod loads the UUT slightly more than the UUT microprocessor. The pod also presents more capacitance than the microprocessor. These effects tend to make any bus drive problems more obvious.

4-45. Clock Loading

The pod increases the normal load on the UUT clock. While this loading will rarely have any effect on clock operation, it may make marginal clock sources more obvious.

4-46. POD DRIVE CAPABILITY

As a driving source on the UUT bus, the pod provides equal to or better than normal 8086 current drive capability. All pod inputs and outputs (except the clock) are TTL-compatible.

4-47. LOW UUT POWER DETECTION

The pod has a UUT power detection circuit which constantly monitors the UUT power supply. If the UUT power supply drops below 4.5V or rises above 5.5V, this circuit produces an output to the troubleshooter which causes the troubleshooter to display a UUT power fail error message.

Also, anytime the UUT power supply drops below about 3.4V, all active pod outputs are disabled or written to their low logic level. This feature has been incorporated to protect UUT circuits from possibly being damaged by pod outputs when the UUT power supply drops below safe operating limits. The troubleshooter will display a pod timeout error message. When the proper operating power supplies have been restored to the UUT, the outputs of the pod will return to normal and the troubleshooter will be ready for additional testing.

Section 5
Theory of Operation

5-1. INTRODUCTION

The theory of operation of the pod is described on two levels. The first level is an overall functional description which describes the major sections of the pod and how they relate to each other and to the UUT and the troubleshooter. The second level is a detailed block diagram of each pod section. The descriptions are supported by block diagrams and timing diagrams in this section and by complete instrument schematics in Section 8 of this manual.

5-2. GENERAL POD OPERATION

The pod is essentially a complete microprocessor system by itself. It is usually in a housekeeping mode, waiting for instructions from the troubleshooter. When the pod receives an instruction, it performs an operation or series of operations on the UUT microprocessor bus, using a bus switch approach. Under normal operating conditions, when the pod is in communication with the troubleshooter, it functions like any normal microprocessor-controlled system. However, when accessing the UUT, the bus is momentarily (for the duration of a memory access cycle or I/O cycle) switched to the UUT by disabling the components in the pod and connecting all lines to the UUT buffered in the appropriate direction.

When the pod emulates the UUT microprocessor in the Run UUT mode, the components within the pod are permanently disabled, and the pod microprocessor is effectively permanently connected to the UUT.

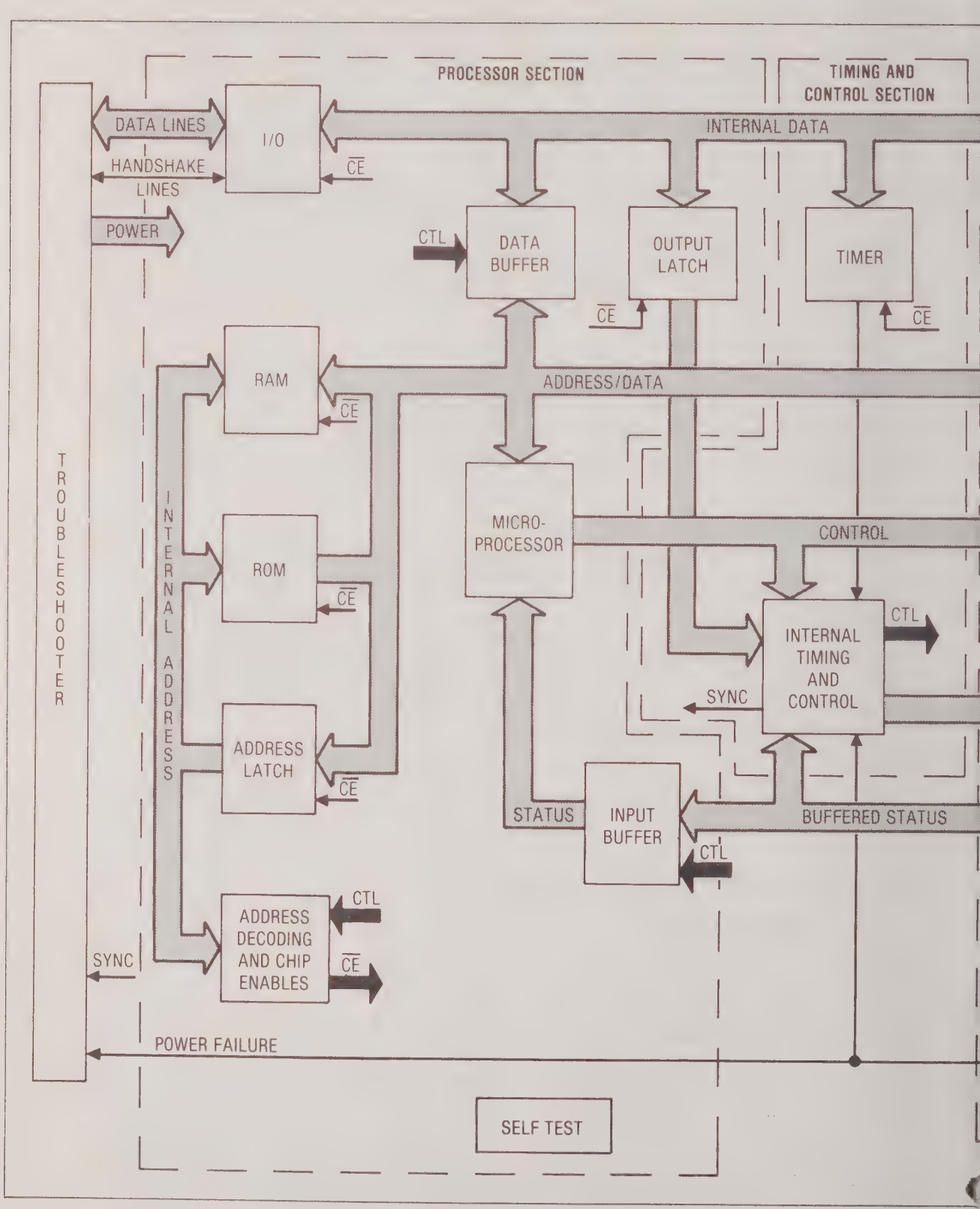
The pod may be divided into three major sections:

- Processor Section
- Timing and Control Section
- Interface Section

Each section is described in the following paragraphs.

5-3. Processor Section

The Processor Section, shown in Figure 5-1, is made up of the microprocessor, RAM, ROM, I/O, and various latches and buffers. These elements, along with some timing signals, comprise a small microsystem which receives commands from the troubleshooter and performs specific operations in response to these commands.


PROCESSOR SECTION
TIMING AND CONTROL SECTION
DATA LINES
HANDSHAKE LINES
POWER
I/O
CE
INTERNAL DATA
CTL
DATA BUFFER
OUTPUT LATCH
TIMER
RAM
ADDRESS/DATA
ROM
MICRO-PROCESSOR
CONTROL
INTERNAL TIMING AND CONTROL
SYNC
ADDRESS LATCH
INTERNAL ADDRESS
STATUS
INPUT BUFFER
BUFFERED STATUS
ADDRESS DECODING AND CHIP ENABLES
POWER FAILURE
SELF TEST
TROUBLESHOOTER

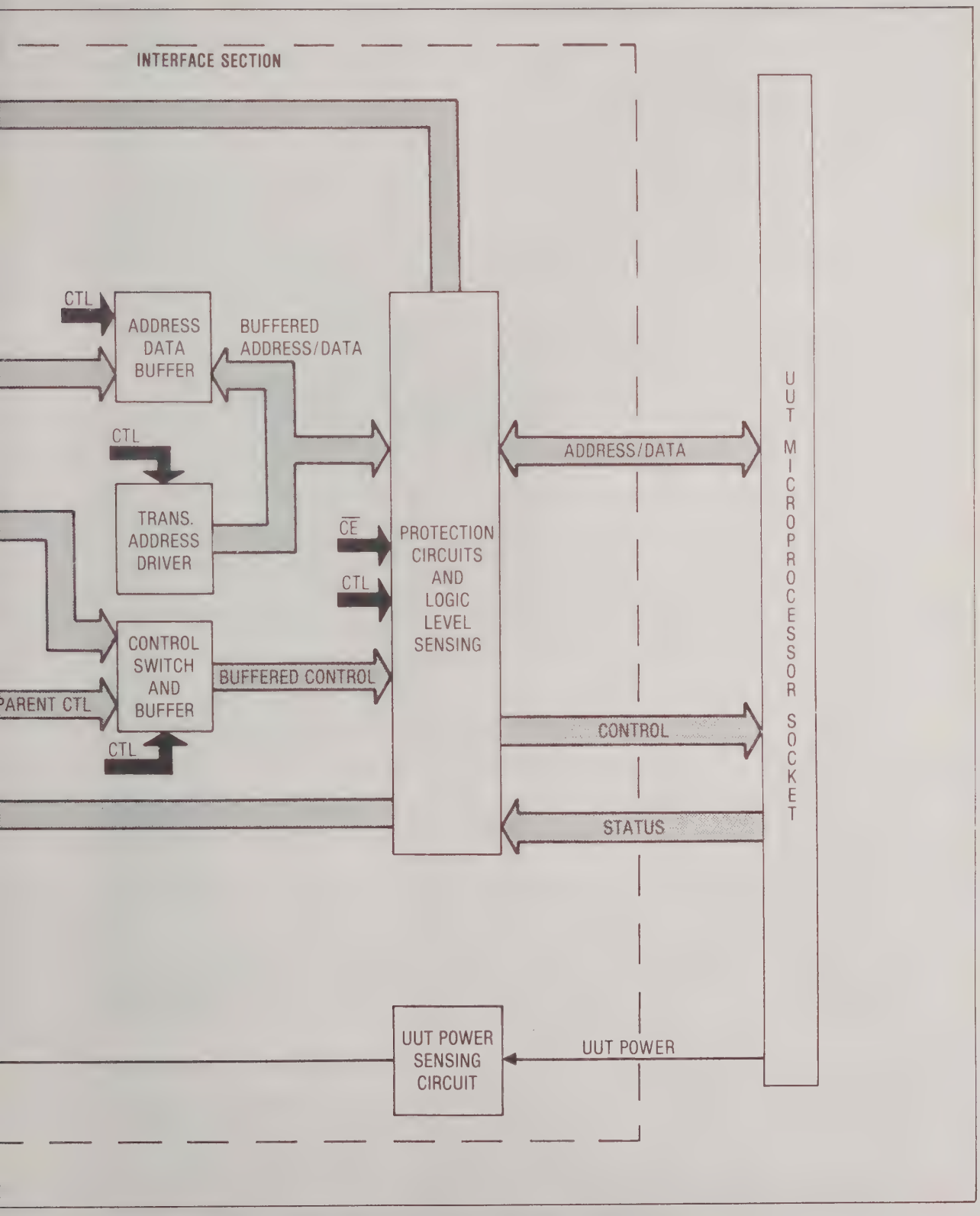


Figure 5-1. 8086 Interface Pod General Block Diagram

For the 8086 pod, as well as all 9000 Series Microsystem pods, the microprocessor inputs are referred to as status lines, and the outputs are referred to as control lines. This nomenclature is not always in agreement with the manufacturer's literature (the 8086 manufacturer, for example, refers to line S3, which is a microprocessor output, as a status line). This convention, however, allows consistency between pods when implementing the troubleshooter functions which involve status or control lines, such as Read Status or Write Control.

All pod status lines which could adversely affect the pod operation are either automatically hardware disabled by the pod, or may be disabled by the operator using the troubleshooter Setup function; the disabling is accomplished with input buffers. Disabling these status lines allows the pod to operate in hostile UUT environments where malfunctioning status lines such as Hold could prevent the pod from performing any tests. The one microprocessor input which may not be disabled, of course, is the UUT clock. The clock signal is buffered for protection purposes, but it must always be present for pod operation. All the status lines are enabled in the Run UUT mode.

The Processor Section also contains circuitry for pod self test. When the pod ribbon cable plug is inserted into the self test socket, part of the pod circuitry becomes a simplified pseudo UUT. During pod self test, certain tests are performed on this pseudo UUT, and any failures are reported to the troubleshooter.

5-4. Timing and Control Section

The Timing and Control Section, shown in Figure 5-1, consists of a timer and internal timing and control logic. The Timing and Control Section receives inputs from the Processor Section and the Interface Section. The bus switch is accomplished by buffer control signals and chip enable signals generated by the Timing and Control Section in response to inputs from the timer, the output latches as set by the microprocessor, the status lines from the UUT, and the control lines from the microprocessor.

The length of a normal bus switch equals one microprocessor memory access cycle or I/O cycle. The bus is switched to communication with the UUT between the last internal operation and the start of the intended UUT operation. The bus switch is initiated by a signal from the timer. The bus is switched back to communication with the troubleshooter at the end of the cycle.

If the microprocessor has sent the Run UUT command through the output latch, the bus switch is started in the normal fashion, but is then held on indefinitely until a reset pulse is received from the troubleshooter.

During the time the pod is not communicating with the UUT, the UUT needs the proper signals so that it can perform the normal refresh operation and other similar tasks. In order to provide these signals to the UUT, the pod performs what are called transparent reads. A transparent read is a read operation that is performed at the reset address. Transparent reads generate the transparent or fake control signals required to simulate a normal microprocessor read operation. This allows UUT operation even when the pod microprocessor is not communicating with the UUT.

5-5. Interface Section

The Interface Section, shown in Figure 5-1, consists of buffers and drivers, protection circuits, logic level detection circuits, and a UUT power sensing circuit. The buffers and drivers are enabled to connect the UUT to the microprocessor or to the transparent control and address signals as dictated by the Timing and Control Section.

Each line to the UUT contains a protection circuit. The protection circuit consists of a 100-ohm resistor in series with a pair of clipping diodes. This circuit prevents overvoltage

or undervoltage conditions from damaging pod components. A 700-ohm resistor in series with the inputs of the CMOS latches of the detection circuits limits the current to protect the internal protection diodes.

The detection circuits consist of latches connected through the 700-ohm protection resistors to the UUT side of the 100-ohm resistors. The latches are latched during a UUTON cycle, when the signals are expected to be at a known level. If a signal cannot be driven through the 100-ohm resistor, it will be detected when the latches are individually read by the Processor Section, and the values are compared with the expected values.

The UUT power sensing circuit shown in Figure 5-1 constantly monitors the UUT power supply. This circuit produces an output to the troubleshooter in the event UUT power drops below 4.5V or rises above 5.5V.

Also, anytime the UUT power supply drops below about 3.4V, all active pod outputs are disabled or written to their low logic level. This feature has been incorporated to protect UUT circuits from being damaged by pod outputs when the UUT power supply drops below safe operating limits. The troubleshooter will display a UUT power-fail error message, or, if the power-fail error message has been disabled, the troubleshooter will indicate a pod timeout error message. When the proper operating power supplies have been restored to the UUT, the outputs of the pod will return to normal, and the troubleshooter will be ready for additional testing.

5-6. DETAILED BLOCK DIAGRAM DESCRIPTION

A detailed block diagram of each major pod section is presented in Figure 5-2. Where possible, the reference designation numbers of specific components are shown in Figure 5-2. The use of the reference designation numbers indicates that most or all of that component is used within the part of the circuit shown. Portions of unlisted components may also be used in the circuit. Each major section is described in the following paragraphs.

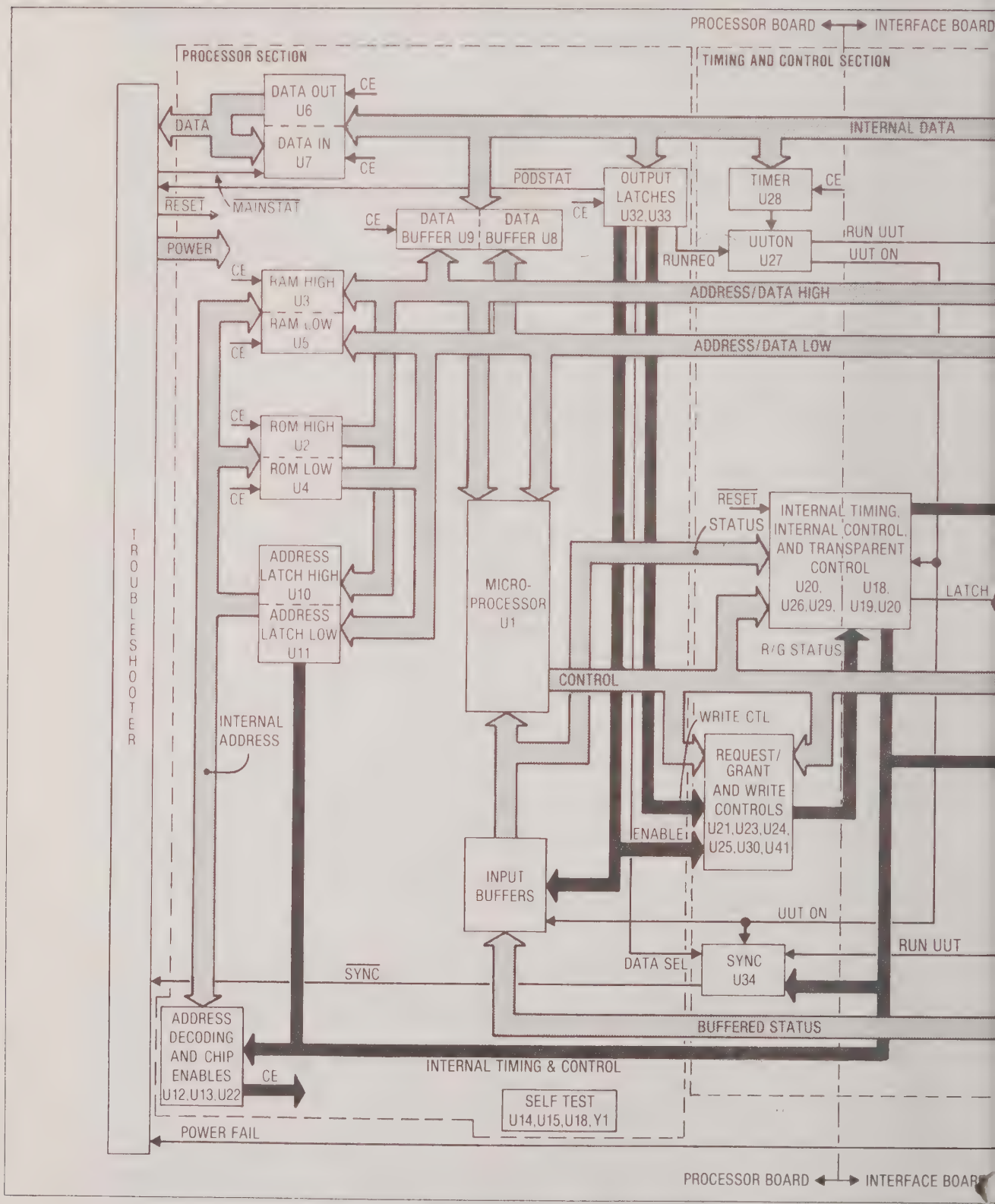
5-7. Detailed Description of Processor Section

The microprocessor (U1), RAMs (U3, U5) and ROMs (U2, U4) comprise a small microprocessor system which is the heart of the pod. Because the address/data bus is time-multiplexed, the address latches (U10, U11) are required to supply valid addresses to the RAM and ROM during the entire memory cycle. The data side of the RAMs and ROMs is tied directly to the address/data bus because the RAMs and ROMs cannot tolerate the extra delay of the data buffers. All other data is input through data buffers (U8, U9) to reduce loading on the bus. The Processor Section communicates with the troubleshooter through additional buffers (U6, U7).

All communication between the pod and the troubleshooter is fully handshaked according to the protocol shown in Figure 5-3. The two handshake lines are $\overline{\text{MAINSTAT}}$ and $\overline{\text{PODSTAT}}$. $\overline{\text{MAINSTAT}}$ is output by the troubleshooter and monitored by the the pod. $\overline{\text{MAINSTAT}}$ initiates all data transactions. $\overline{\text{PODSTAT}}$ indicates the pod response.

The address decoding and chip enable circuitry (U12, U13, U22) select which components or buffers are enabled on the data bus. This circuitry also controls the direction of the data buffers, and disable all internal components during UUTON.

The output latches (U32, U33) are used by the microprocessor to control the function of the pod. The output latches control which inputs are enabled, the value of lines during a Write Control operation, which sync pulse is returned to the troubleshooter, and the Run UUT mode. $\overline{\text{PODSTAT}}$ is also one of the outputs of the latches.


PROCESSOR BOARD
INTERFACE BOARD
PROCESSOR SECTION
TIMING AND CONTROL SECTION
DATA OUT U6
DATA IN U7
CE
DATA
INTERNAL DATA
PODSTAT
OUTPUT LATCHES U32,U33
TIMER U28
RESET
MAINSTAT
DATA BUFFER U9
DATA BUFFER U8
RUNREQ
UUTON U27
RUN UUT
UUT ON
POWER
RAM HIGH U3
RAM LOW U5
ADDRESS/DATA HIGH
ADDRESS/DATA LOW
ROM HIGH U2
ROM LOW U4
ADDRESS LATCH HIGH U10
ADDRESS LATCH LOW U11
MICRO-PROCESSOR U1
INTERNAL TIMING, INTERNAL CONTROL, AND TRANSPARENT CONTROL U20, U26,U29, U18, U19,U20
STATUS
LATCH
R/G STATUS
CONTROL
WRITE CTL
INTERNAL ADDRESS
REQUEST/ GRANT AND WRITE CONTROLS U21,U23,U24, U25,U30,U41
ENABLE
INPUT BUFFERS
UUT ON
RUN UUT
DATA SEL
SYNC U34
SYNC
BUFFERED STATUS
ADDRESS DECODING AND CHIP ENABLES U12,U13,U22
INTERNAL TIMING & CONTROL
SELF TEST U14,U15,U18,Y1
POWER FAIL
TROUBLESHOOTER
PROCESSOR BOARD
INTERFACE BOARD

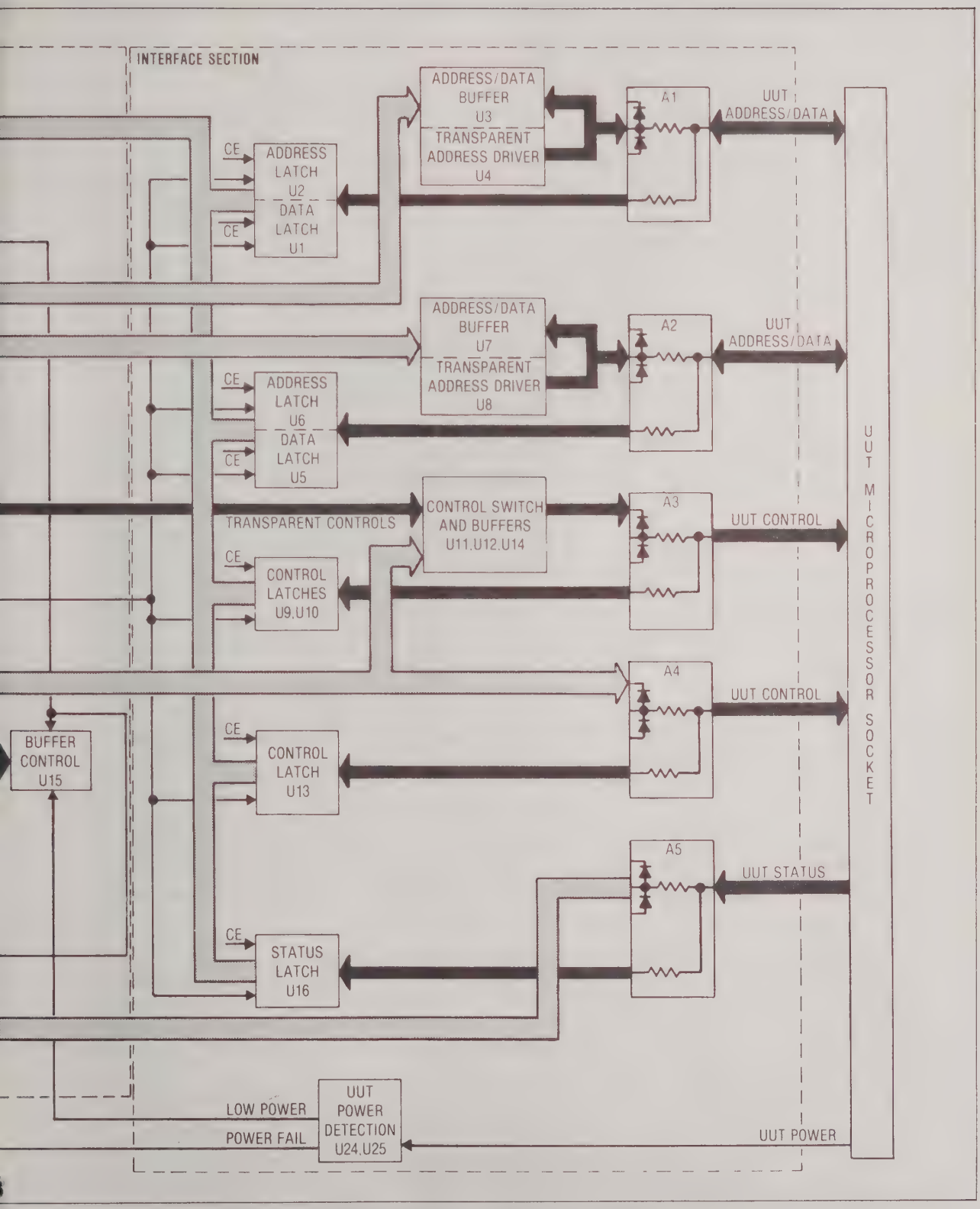


Figure 5-2. 8086 Interface Pod Detailed Block Diagram

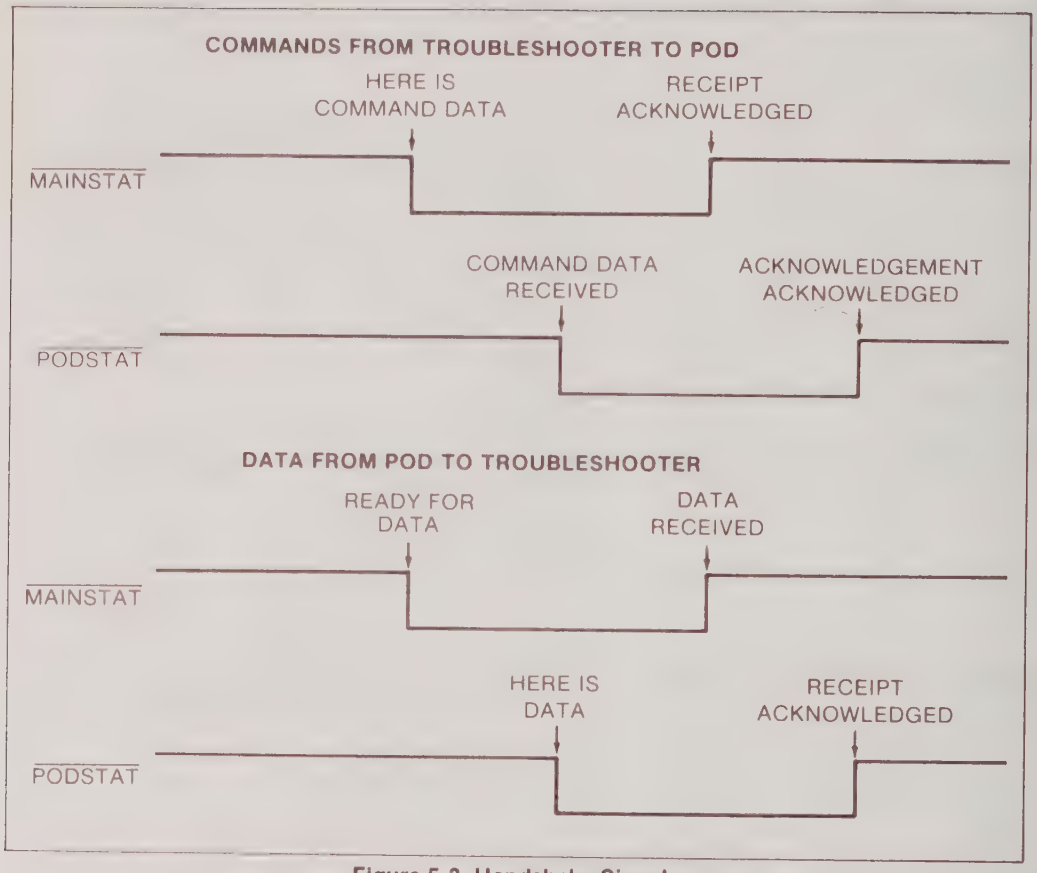


Figure 5-3. Handshake Signals

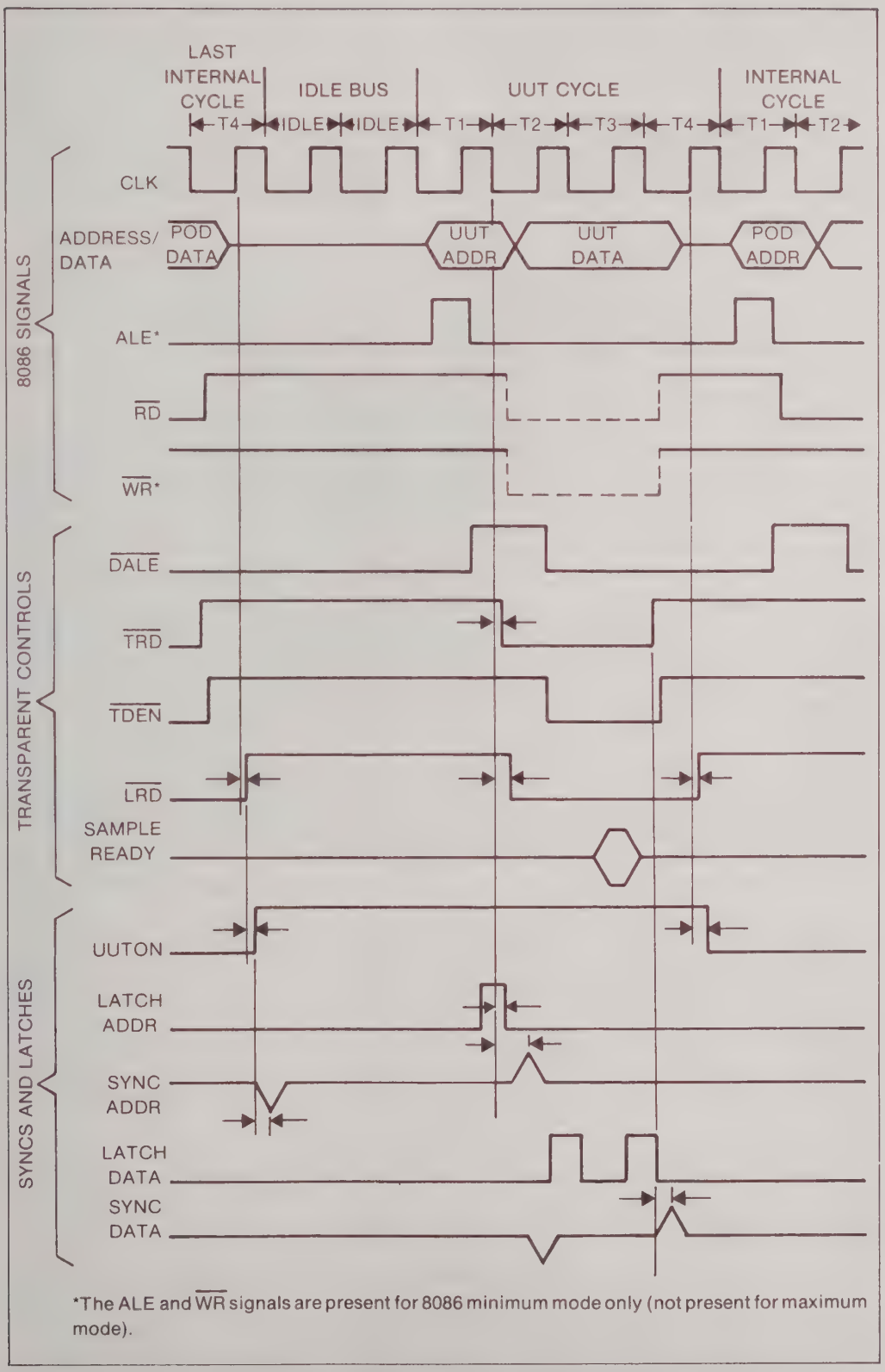
The various input buffers help protect the microprocessor from potentially damaging signal levels at the UUT and allow or disallow inputs, depending on which lines are enabled. The input buffers allow all inputs if the Run UUT mode is selected.

The self test circuitry consists of a clock and clock buffer (Y1, U18) and some latches which latch the address (U14, U15). The outputs of these latches are placed on the data bus during a read operation. All status lines are tied to the active state to ensure that the pod can operate in the potentially hostile UUT environment. Vcc is returned to check the power level.

5-8. Detailed Description of Timing and Control Section

The components that comprise the Timing and Control Section are located on both the processor and interface boards. The Timing and Control Section is divided into functional subsections for the purposes of discussion, although some of the subsections are intricately interrelated.

The timing relationships of several important pod signals is shown in Figure 5-4. The signals include UUTON, the latch controls, and sync pulse. The UUTON circuit (U27 and associated logic gates) controls the bus switch timing and enable signals. The UUTON signal is initiated by an output of the timer (U28), and is normally ended by the signal $\overline{\text{LRD}}$ which is generated by the internal timing circuits. $\overline{\text{LRD}}$ is an extended or long read pulse. If RUNREQ is on (high), UUTON stays on (high) until a reset is received from the troubleshooter.



*The ALE and WR signals are present for 8086 minimum mode only (not present for maximum mode).

Figure 5-4. 8086 Signal Timing Relationships

The sync circuitry (U34 and associated gates) sends a sync signal to the troubleshooter during the address or data portion of UUTON. (Address or data sync is selected by the Processor Section.) Sync is inhibited by the signal RUNUUT to prevent damage to the probe pulser.

The buffer control circuitry (U15) controls the bus switch by controlling the buffer direction and enable signals of the buffers in the Interface Section. Component U15 is a PAL (Programable Array of Logic); a schematic diagram of U15 is included in Section 8 of this manual.

The internal timing, internal control, and transparent control circuitry is the group of circuits which is used to generate a stable set of usable control signals for both the minimum and the maximum mode of 8086 operation. Most of the signals generated have names that are similar to the names of the microprocessor control lines, with an I or a T added to the name to indicate whether the signal is used internally or is transparent. The internal signals are signals that are not sent to the UUT, but are used solely for internal pod operations. The transparent signals are signals which are sent to the UUT during a transparent read. (Transparent reads are described in the previous section titled Timing and Control Section.) Some of the transparent signals are also used internally. U20 latches the $\overline{BHE}$ and DT/$\overline{R}$ signals for internal use. U29 is used to generate interrupt acknowledge and ALE in the maximum mode, and U26 is used to switch between minimum and maximum mode signals.

As part of the internal timing, internal control, and transparent control circuitry on the interface board, U19 stretches the IALE pulse and generates the $\overline{TDEN}$ signal. U18 generates a TRD signal, and stretches $\overline{INTA}$ over both cycles. U20 generates the $\overline{LRD}$ (Long RD) signal. U21 stretches the PRESET pulse to keep the buffers tristated for two extra clock cycles, emulating the operation of the microprocessor.

The Request/Grant circuitry, which is shown in Figure 5-5, keeps track of the sequential RQ/GT pulses in the maximum mode. U23 is a sequential logic PAL (Programmable Array Logic) that monitors the RQ/GT pulses, pod and UUT resets, enables, minimum or maximum mode, and Write Controls. The U23 outputs are defined as follows:

SIGNALS	DESCRIPTION
HOLD0, HOLD1	A request pulse has been received.
HLDA0, HLDA1	The pod has granted the bus.
UHLDA0, UHLDA1	The UUT has not released the bus.
FAKE0, FAKE1	A fake request should be sent to the pod microprocessor after enable, or a fake release.

U24 and U25 control the buffer enables and signal directions on these lines in response to U23 and enable signals. Some of the logic gates in U24 and U25 are used to write the other control lines. Schematics for U24 and U25 are provided in Section 8.

U41 receives the UHLDA and FAKE outputs from U23. The DHLDA and DFAKE outputs of U41 are the inputs delayed by one clock cycle. The $\overline{RQGT}$ outputs are pulsed low for one clock cycle at the trailing edge of the FAKE input (which is the fake request or fake release to the pod microprocessor). The PBRQ outputs are the pulse buffered requests (RQ). The PBRQ signals enable U14 on the interface board which acts as an

active pullup on the UUT RQ/GT line for one clock cycle on every positive or negative transition of UHLDA.

5-9. Interface Section

Components A1 through A5 are Fluke-designed hybrid ICs containing current-limiting resistors and clipping diodes to protect pod circuits from overvoltage conditions. A1 through A5 also contain (not shown) 200-kilohm pull-up resistors for the inputs of the latches.

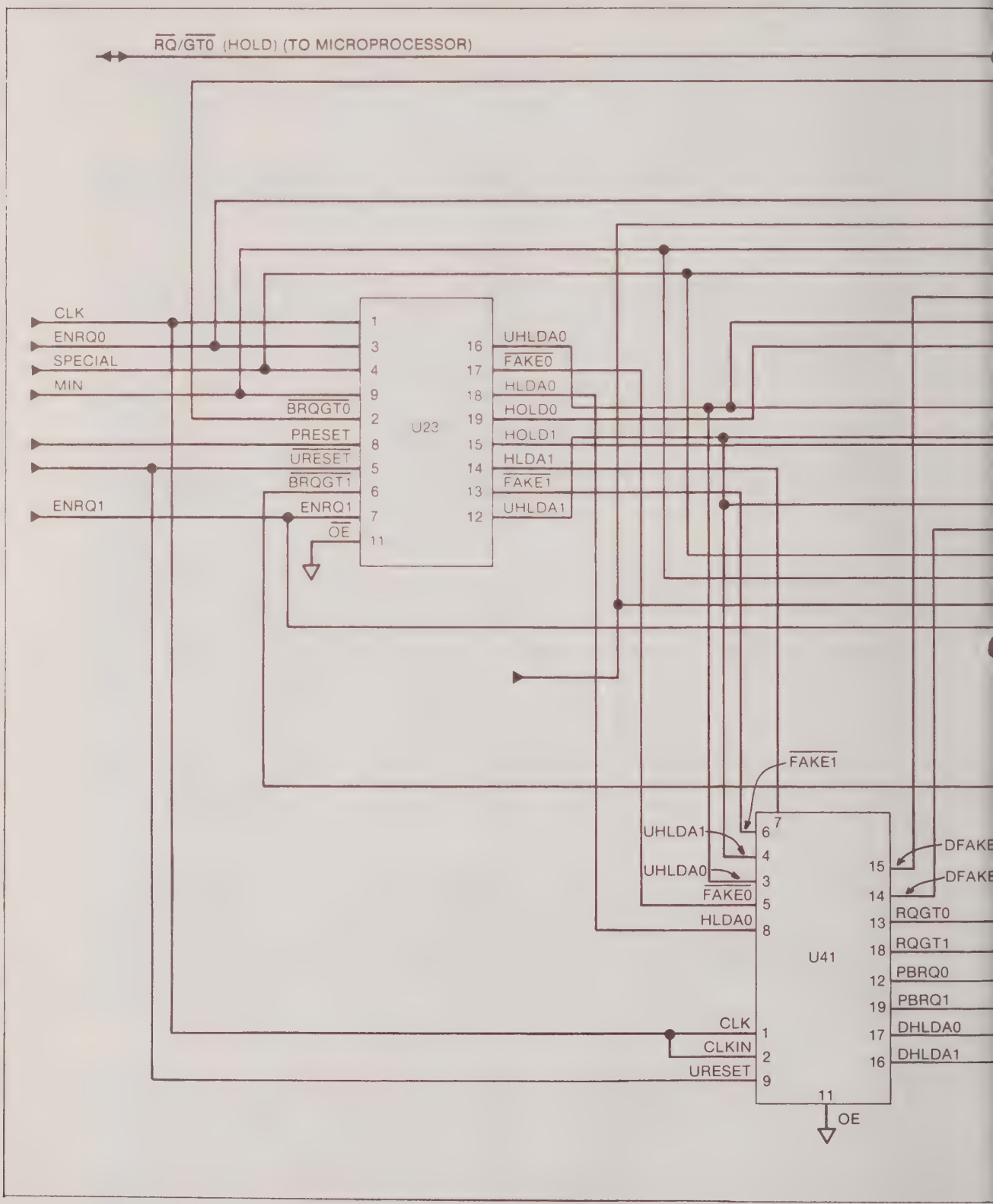
The enabling and the direction of the address/data buffers (U3, U7) is controlled by the Timing and Control Section. The address/data buffers are turned on when the Processor Section is communicating with the UUT. The transparent address drivers (U4, U8) are enabled during the address portion of transparent reads. Both buffers and drivers are disabled during a Hold sequence, during a reset, or if UUT power drops below approximately 3.5V.

The control switch and buffer circuitry (U11, U12, U14) performs the same function for the control lines as the address/data buffers and transparent address drivers perform on the address/data line (i.e., drive the chosen control signals to the UUT or tristate when required). The control switch and buffer circuitry is under the direction of the Timing and Control Section.

The inputs to all the latches (U1, U2, U5, U6, U9, U10, U13, U16) are connected to the UUT lines through the 700-ohm resistors of the components A1 through A5. The latches are latched under control of the Timing and Control Section during a UUT access or during an interrupt acknowledge cycle. The latches may receive multiple latch pulses during a UUT cycle; the last data latched during the UUT cycle is the data read by the Processor Section. (The Processor Section compares the latched data with what the Processor Section expected to see to detect drivability errors, forcing lines pending, or interrupts pending.)

The UUT power detection circuit constantly monitors the UUT power supply. If the UUT power supply drops below 4.5V or rises above 5.5V, the power detection circuit produces an output to the troubleshooter which causes the troubleshooter to display a UUT power fail error message.

Also, anytime the UUT power supply drops below about 3.4V, all active pod outputs are disabled or written to their low logic level. (This feature has been incorporated to protect UUT circuits from being damaged by pod outputs when the UUT power supply drops below safe operating limits.) If this happens, the troubleshooter displays a pod timeout error message. When the proper operating power supplies have been restored to the UUT, the outputs of the pod return to normal and the troubleshooter is ready for additional testing.


RQ/GT0 (HOLD) (TO MICROPROCESSOR)
CLK
ENRQ0
SPECIAL
MIN
BRQGT0
PRESET
URESET
BRQGT1
ENRQ1
ENRQ1
OE
U23
UHLDA0
FAKE0
HLDA0
HOLD0
HOLD1
HLDA1
FAKE1
UHLDA1
FAKE1
UHLDA1
UHLDA0
FAKE0
HLDA0
CLK
CLKIN
URESET
U41
DFAKE
DFAKE
RQGT0
RQGT1
PBRQ0
PBRQ1
DHLDA0
DHLDA1
OE

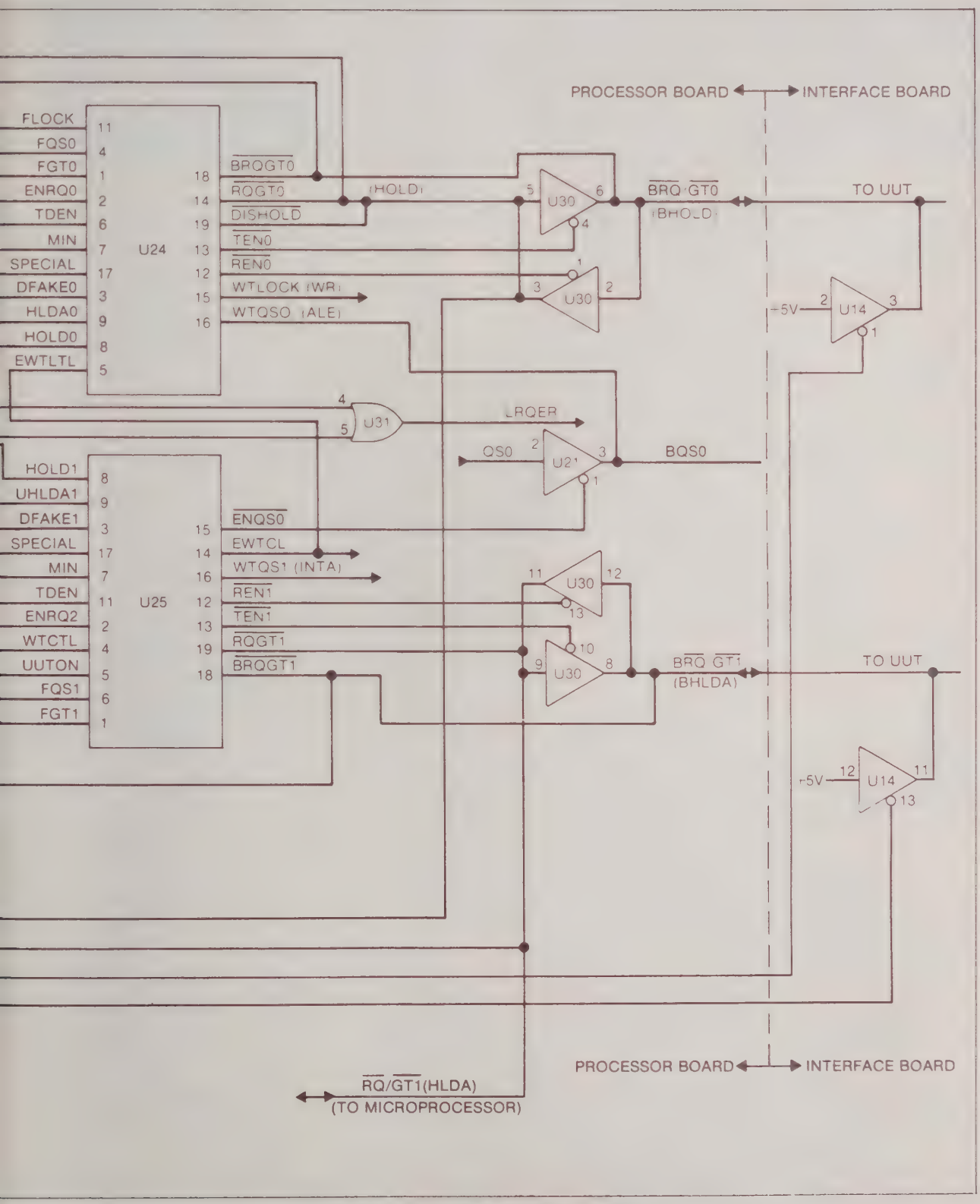
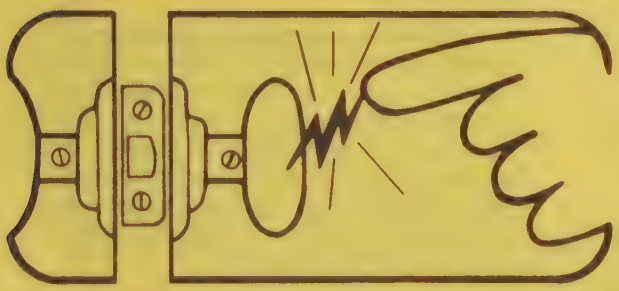


Figure 5-5. The Request/Grant Circuitry

static awareness

A Message From

John Fluke Mfg. Co., Inc.



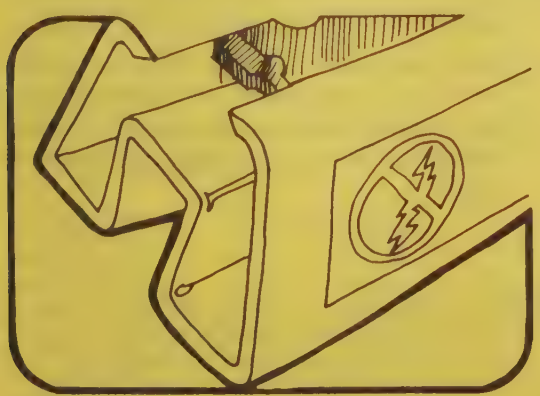
Some semiconductors and custom IC's can be damaged by electrostatic discharge during handling. This notice explains how you can minimize the chances of destroying such devices by:

1. Knowing that there is a problem.
2. Learning the guidelines for handling them.
3. Using the procedures, and packaging and bench techniques that are recommended.

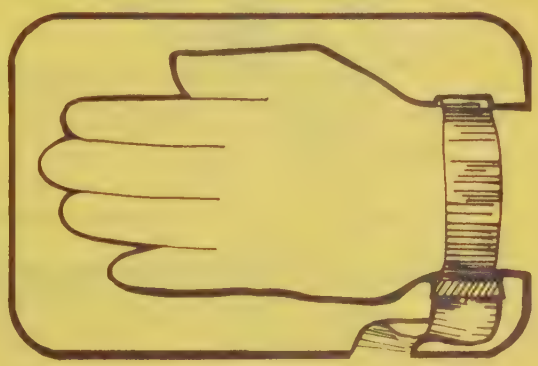
The Static Sensitive (S.S.) devices are identified in the Fluke technical manual parts list with the symbol "⊗"

The following practices should be followed to minimize damage to S.S. devices.

1. MINIMIZE HANDLING

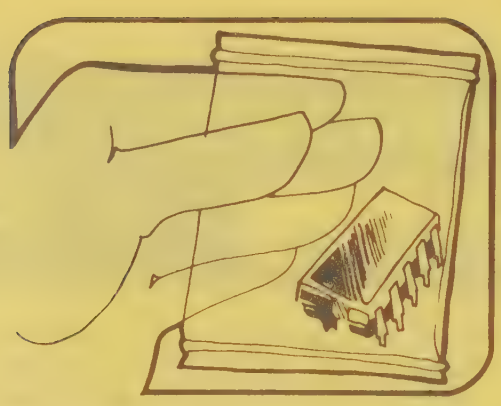


2. KEEP PARTS IN ORIGINAL CONTAINERS UNTIL READY FOR USE.

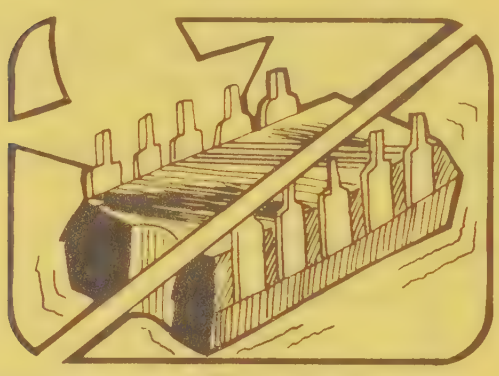


3. DISCHARGE PERSONAL STATIC BEFORE HANDLING DEVICES

4. HANDLE S.S. DEVICES BY THE BODY

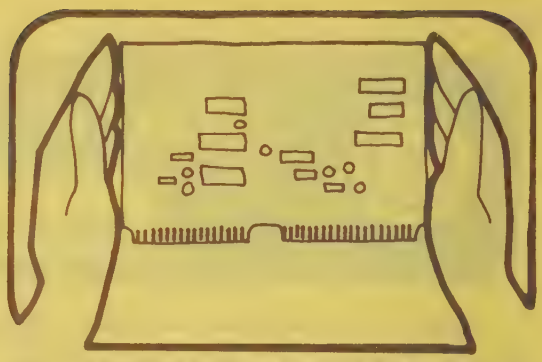


5. USE ANTI-STATIC CONTAINERS FOR HANDLING AND TRANSPORT

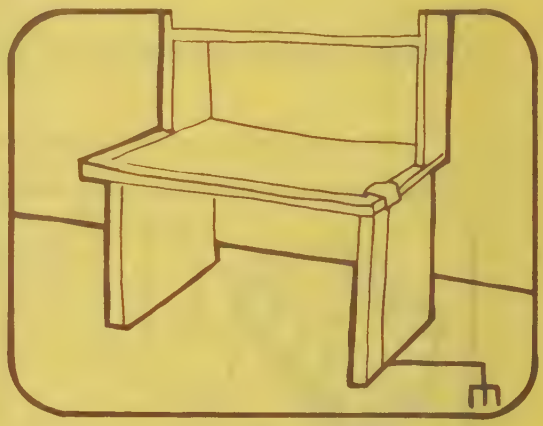


6. DO NOT SLIDE S.S. DEVICES OVER ANY SURFACE

7. AVOID PLASTIC, VINYL AND STYROFOAM® IN WORK AREA



8. WHEN REMOVING PLUG-IN ASSEMBLIES, HANDLE ONLY BY NON-CONDUCTIVE EDGES AND NEVER TOUCH OPEN EDGE CONNECTOR EXCEPT AT STATIC-FREE WORK STATION. PLACING SHORTING STRIPS ON EDGE CONNECTOR USUALLY PROVIDES COMPLETE PROTECTION TO INSTALLED SS DEVICES.



9. HANDLE S.S. DEVICES ONLY AT A STATIC-FREE WORK STATION
10. ONLY ANTI-STATIC TYPE SOLDER-SUCKERS SHOULD BE USED.
11. ONLY GROUNDED TIP SOLDERING IRONS SHOULD BE USED.

Anti-static bags, for storing S.S. devices or pcbs with these devices on them, can be ordered from the John Fluke Mfg. Co., Inc.. See section 5 in any Fluke technical manual for ordering instructions. Use the following part numbers when ordering these special bags.

John Fluke Part No.	Description
453522	6" X 8" Bag
453530	8" X 12" Bag
453548	16" X 24" Bag
454025	12" X 15" Bag
Pink Poly Sheet 30"x60"x60 Mil P/N RC-AS-1200 $20.00	Wrist Strap P/N TL6-60 $7.00

PORTIONS REPRINTED
WITH PERMISSION FROM TEKTRONIX, INC.
AND GENERAL DYNAMICS, POMONA DIV.

® Dow Chemical

J0089B-07U7810/SE EN Litho in U.S.A.

Section 6
Troubleshooting

WARNING

THESE INSTRUCTIONS ARE FOR USE BY QUALIFIED PERSONNEL ONLY. TO AVOID ELECTRIC SHOCK, DO NOT PERFORM ANY INSTRUCTIONS UNLESS YOU ARE QUALIFIED TO DO SO.

6-1. INTRODUCTION

This section provides troubleshooting information for the pod, and includes repair precautions and disassembly procedures.

The troubleshooting guidelines presented in this section are intended to assist in the isolation of faults within the pod. If you do not want to service the pod yourself, or if attempted troubleshooting fails to reveal the pod fault, you may ship the pod to the nearest Fluke Technical Service Center for repair. If requested, a free cost estimate will be provided before any repair work is performed. Refer to the troubleshooter operator manual or service manual for a list of Fluke Technical Service Centers.

If pod shipment is necessary, the pod should be shipped in its original shipping container if it is available. If the original shipping container is not available, you may order a new container from John Fluke Mfg. Co., Inc.; P.O. Box C9090, Everett, WA 98206; telephone (206) 342-6300.

Troubleshooting the pod is similar to troubleshooting any other microprocessor-based UUT, and requires the equipment listed in Table 6-1. The troubleshooting procedures provided in the following sections are supported by the theory of operation in Section 5 and the schematic diagrams in Section 8.

NOTE

All references to data and addresses in the following sections are in hexadecimal notation.

CAUTION

Static discharge can damage MOS components contained in the pod. To prevent this possibility, take the following precautions when troubleshooting and/or repairing the unit.

- Never remove, install, or otherwise connect or disconnect pcb (printed circuit board) assemblies without disconnecting the pod from the troubleshooter.

Table 6-1. Required Test Equipment for Pod Troubleshooting

EQUIPMENT TYPE	REQUIRED TYPE
Micro System Troubleshooter	Fluke 9000 Series
Interface Pod	Fluke 9000A-8086
Digital Multimeter	Fluke 8020
Oscilloscope	Tektronix 485 or equivalent

- Perform all repairs at a static-free work station.
- Do not handle ICs or pcb assemblies by their connectors.
- Wear a static ground strap when performing repair work.
- Use conductive foam to store replacement or removed ICs.
- Remove all plastic, vinyl, and styrofoam from the work area.
- Use a grounded soldering iron with a rating of 25 watts or less to prevent overheating the pcb assembly.
- When shipping the pod, always place the pod in a static-free plastic bag.

6-2. DETERMINING WHETHER THE POD IS DEFECTIVE OR INOPERATIVE

The first task of troubleshooting the pod is to determine whether the pod is defective or inoperative. This determination is based on the results of the pod self test described in Section 2. If you have not performed the self test, refer to Section 2 and perform the self test before proceeding with the troubleshooting.

Depending on the results of the pod self test and the pod behavior when connected to a known good UUT, the pod may be categorized in one of the three following groups:

- Defective Pod: The pod fails the pod self test and the troubleshooter displays a self test failure code. Refer to the section titled Troubleshooting a Defective Pod.
- Inoperative Pod: The pod is unable to complete the pod self test and the troubleshooter displays an *ATTEMPTING RESET* message. Refer to the section titled Troubleshooting an Inoperative Pod.
- Suspected Defective Pod: The pod passes the pod self test but exhibits abnormal behavior when connected to a known good UUT. Refer to the section titled Extended Troubleshooting Procedures.

6-3. TROUBLESHOOTING A DEFECTIVE POD

This section tells what to do if the troubleshooter displays the following message when the pod self test is performed: *POD SELF-TEST 8086 FAIL xx*. The letters *xx* represent one of the self test failure codes listed in Table 6-2. The *FAIL* message indicates that although there is a problem with the pod, the pod is still communicating with the troubleshooter and is able to help diagnose the problem.

Table 6-2. Standard Self Test Failure Codes

FAILURE CODE	DESCRIPTION
00*	UUT read access failed or the enhanced self test failed*
01	UUT write access failed
02	Control line(s) cannot be driven
03	Enableable status line(s) failed

* When the failure code 00 is received, the pod may have failed an internal enhanced self test. To determine whether the pod failed the enhanced self test, perform a read at F0 00F0. If the data returned is 00FF, the pod did not fail the enhanced self test. If the message returned is anything other than *READ @ F0 00F0 = 00FF OK*, the enhanced self test failed; refer to the section titled Recreating the Enhanced Self Test Routines.

In order to understand how to interpret the self test failure codes, it is important to understand what happens when the pod self test is performed. Whenever Pin 20 of the pod self test socket is connected to ground, the pod senses that it is in self test. Inserting the pod ribbon cable plug into the self test socket connects Pin 20 to ground. The pod self test does not actually begin until the operator attempts an operation involving the pod (such as a Bus Test), and the pod informs the troubleshooter that it is in self test.

When the pod self test is performed, the pod appears to the troubleshooter to be a small, known UUT. The troubleshooter exercises the pod by performing a series of read and write operations. The address for each write operation is latched, then manipulated and returned to the troubleshooter on the data lines. The self test socket provides the power and clock signal for pod operation during self test. All inputs except the clock and power are held in the active state to determine if the pod can operate in a hostile enviroment.

The failure codes for the standard pod self test are listed in Table 6-2. Failure codes 00 through 03 may be detected by any of the 9005A, 9010A, or 9020A Micro-System Troubleshooters. The 9020A performs a more thorough pod self test than either the 9005A or 9010A and may report more failure codes than those listed in Table 6-2. The additional failure codes are listed and described in the 9020A Operator Manual.

As indicated in Table 6-2, the failure code 00 may mean that the pod failed an enhanced self test that is internal to the pod. In addition to the test routines that are performed on the pod by the troubleshooter in the standard self test routine, the pod has an internal enhanced self test which performs some more thorough test routines. In most cases, if the pod has an error, the enhanced self test, which is performed before the standard self test and is more comprehensive, will find the error.

The recommended method for troubleshooting a defective pod is to recreate the self test routine that detected the pod failure and use the routine as the starting point for tracing the problem. The following paragraphs describe how to prepare for troubleshooting, how to determine which self test routine failed, and how to recreate the self test routine.

6-4. Preparation for Troubleshooting a Defective Pod

Prepare to troubleshoot your defective pod as follows:

1. Refer to the later section titled Disassembly, and disassemble the pod.

2. Look for any obvious problems such as burned components or ICs that are loose in their sockets. Replace components if necessary.

3. Plug the pod ribbon cable into the self test socket and lock the socket. Press the BUS TEST key to initiate the self test and then press the STOP key. Perform a *WRITE @ F0 00F0 0;* this disables the pod self test by preventing the pod from reporting to the troubleshooter that it is plugged into itself. (The pod self test may be reenabled by turning the troubleshooter power off and then on again, or by performing a *WRITE @ F0 00F0 = XX* where *XX* is any non-zero value.)

NOTE

When the pod is prevented from reporting self test to the troubleshooter, the pod may be used to troubleshoot itself. The method used with other pods to disable self test is to disconnect the ground pin on the self test socket. This method cannot used with the 8086 pod because this would disable the pod self test clock driver. Remember that in order to perform the pod self test, you must reenable the pod self test by performing a power up reset or by a WRITE @ F0 00F0 = xx where xx does not equal 0.

4. Press the SETUP key on the troubleshooter and set the following conditions:

SET-TRAP BAD POWER SUPPLY? YES
SET-TRAP ILLEGAL ADDRESS? NO
SET-TRAP ACTIVE INTERRUPT? NO
SET-TRAP ACTIVE FORCE LINE? NO
SET-TRAP CTL ERR? YES
SET-TRAP ADDR ERR? YES
SET-TRAP DATA ERR? YES
SET-ENABLE READY? NO
SET-ENABLE HOLD? NO
SET-ENABLE INTR? NO

You are now ready to explore the possible causes of pod failure that are indicated by the pod self test failure codes.

Whenever the pod self test is performed and the troubleshooter displays the message *POD SELF TEST 8086 FAIL xx* where *xx* equals 01, 02, or 03, the pod failed the standard self test. Refer to the section titled Recreating the Standard Self Test Routines.

Whenever the pod self test is performed and the troubleshooter displays the message *POD SELF TEST 8086 FAIL 00*, the pod may have failed either the standard self test or the enhanced self test. In most cases the enhanced self test, which is more thorough, will detect the failure.

To find out if the pod failed the enhanced self test or the standard self test, perform a read operation at F0 00F0. If the resulting message is *READ @ F0 00F0 00FF OK*, then the pod passed the enhanced self test, which implies that any reported failures were detected by the standard self test. Refer to the section titled Recreating the Standard Self Test Routines.

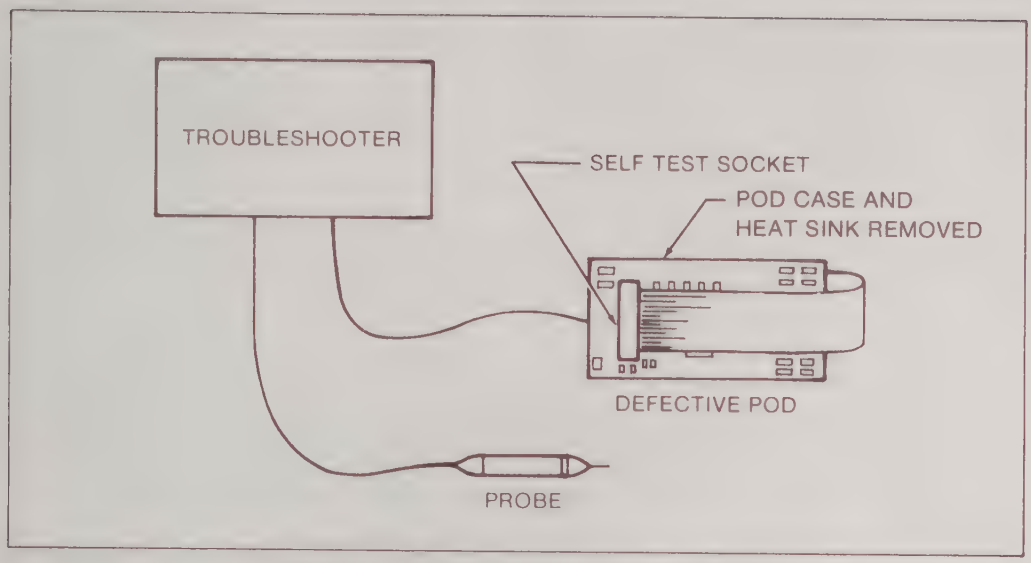


Figure 6-1. Troubleshooting a Defective Pod

If the message resulting from a read operation at F0 00F0 is anything other than *READ @ F0 00F0 = 00FF OK*, then the pod failed the enhanced self test. Proceed to the next section titled Recreating the Enhanced Self Test Routines for appropriate troubleshooting procedures.

6-5. Recreating the Enhanced Self Test Routines

The enhanced self test consists of nine test routines that are performed on the pod circuitry. The test routines are listed and described in Table 6-3.

Whenever the pod fails the enhanced self test, the pod causes the troubleshooter to display the message *POD SELF TEST 8086 FAIL 00*. To confirm that the enhanced self test failed, perform a read operation at address F0 00F0. If the troubleshooter displays any message other than *READ @ F0 00F0 = 00FF OK*, then the pod failed the enhanced self test. (In many cases an error message will be displayed.) Press the MORE key to find out more information about the error that was detected. (Sometimes more than one line of information is available.) Then press the CONT key to find out which enhanced self test routine failed.

For example, assume the pod self test is performed and the troubleshooter displays the message *POD SELF TEST 8086 FAIL 00*. The following sequence of keystrokes provides information about the pod failure detected:

PRESS	DISPLAY	COMMENT
READ F000F0 ENTER	*DATA ERR @ F0 00F0-LOOP?*	Enhanced self test detected data error.
MORE	*DATA BITS 0080—LOOP?*	Unexpected data on data bit 7 (0080 corresponds to bit 7).
CONT	*READ @ F0 00F0 = 0002 FAIL*	The failure occurred during test routine 2.

Table 6-3. Enhanced Self Test Failure Codes

FAILURE CODE/ TEST ROUTINE	OPERATION PERFORMED BY POD	EXPECTED DATA (READ OPERATIONS ONLY)
0000	READ @ 0A AAAA	ABAA
0001	WRITE @ 1A AAAA = 5555	
0002	READ @ A5 5555	0054
0003	READ @ A5 5556	0055
0004	WRITE @ B5 5556 = 00AA	
0005	READ @ 40 CCCC	CDCC
0006	WRITE @ C0 3333 = 00CC	
0007	WRITE @ CTL = 1 0000	
0008	WRITE @ CTL = 0 0100	
00FF	No failures were detected	

NOTE

If you press the LOOP key or the YES key in response to the LOOP? prompts which accompany the preceding messages, the troubleshooter loops on the READ @ F0 00F0; the troubleshooter does not loop on the test routine in which the failure occurred.

The previous sequence of messages indicates that data bit 7 was found to be in error while the pod was performing test routine 2. Note that although a data error implies a drivability error, when encountered with the enhanced self test a data error may also indicate that incorrect data was received during a read operation.

To recreate the self test routine, make sure the pod is prepared for troubleshooting by performing the steps listed in the previous section titled Preparation for Troubleshooting a Defective Pod (which includes disabling the pod self test detection). Then perform the operation listed for test routine 2 in Table 6-3, which is a read operation at address A5 5555. If the pod is operating correctly, the resulting data will be 0054. However, since the pod reported that bit 7 was incorrect, the data will probably be 00D5. Use a synchronized oscilloscope or synchronized probe and loop on the operation to trace the problem.

If you are using an oscilloscope, use the Quick Looping Read described in Section 4 so that the signal is easier to view on the oscilloscope; the Quick Looping Read at address A5 5555 is initiated by performing a read operation at 1A5 5555.

The other enhanced self test routines may be recreated in a similar manner.

6-6. Recreating the Standard Self Test Routines

Most of the problems that occur with pod circuitry will be detected by the enhanced pod self test. However, there are a few problems that may be detected only by the standard pod self test. If the pod fails the standard pod self test, the recommended procedure is the same as the recommended procedure for failure of the enhanced self test. You may recreate the standard self test routine that detected the failure and use that routine as the starting point for subsequent troubleshooting. The failure codes for the standard self test are listed in Table 6-3.

If the failure code is 00, perform a read operation at 0FF0 0FF0 and observe the error message. If the error message indicates power fail, check the power detection circuits. If the error message indicates control line or address line drivability errors, press the MORE key to scroll to the second line of the error message to find out which microprocessor line or lines are not drivable. Then use a synchronized oscilloscope (with the Quick Looping Read) or a synchronized probe and loop on the error to locate the problem. If no error messages occur, check the data for an error (the data should be F00F).

If the failure code is 01, perform a *WRITE @ 0FF0 0FF0 = 0FF0.* Follow the same procedures as described for failure code 00.

If the failure code is 02, perform a Bus Test. If you get a control drivability error, try to find a single troubleshooting operation (rather than the Bus Test) which produces the same error. (The synchronized probe or scope is not useful with the Bus Test which performs many UUT accesses; however, the synchronized probe or scope is useful when used with a single troubleshooting operation.) Try a Write Control if the line is writable. Try I/O reads and writes, and reads and writes to odd bytes.

If the failure code is 03, one of the enables failed to disable the pod. During the pod self test, each one of the pod enable lines are enabled in turn to check whether the pod times out as it should when an enable line is enabled. (These pod timeouts are not visible to the operator when they occur during the pod self test.) If a pod timeout fails to occur for each enable line (except for enable line INTR), the troubleshooter displays failure code 03.

To check the enable lines, use the troubleshooter Setup function and enable each enable line separately. After enabling a line, perform a troubleshooter operation that involves the pod, such as a read operation. If a pod timeout fails to occur, something is probably preventing the enable line signal from getting to the microprocessor.

The same basic techniques may be used for the additional failure codes that may be obtained when using the 9020A Micro-System Troubleshooter.

6-7. TROUBLESHOOTING AN INOPERATIVE POD

6-8. Introduction

This section describes what to do if the troubleshooter displays any of the three *ATTEMPTING RESET* messages when the pod self test is performed. The *ATTEMPTING RESET* messages indicate that the pod is not operating and is not responding to the troubleshooter.

If you correct a problem while using the procedures provided in this section, try the pod self test again. If the troubleshooter again displays an *ATTEMPTING RESET* message, continue with the procedures in this section. However, if the troubleshooter displays the message *POD SELF-TEST 8086 FAIL xx,* refer to the previous section titled Troubleshooting a Defective Pod. The reason for referring to the other section is that when the pod is again communicating with the troubleshooter, you may use the pod to help troubleshoot itself.

6-9. Procedure for Troubleshooting an Inoperative Pod

An inoperative pod is like any other microprocessor-based UUT that is not operating properly; the easiest way to fix an inoperative pod is by using a troubleshooter and a good pod. Perform the following steps to prepare for troubleshooting:

1. Refer to the later section titled Disassembly, and disassemble the pod.

2. Look for any obvious problems such as burned components or ICs that are loose in their sockets. Replace components if necessary.

3. Remove the pod microprocessor from its socket and install the ribbon cable plug from the good pod in the microprocessor socket (as shown in Figure 6-2).

4. To provide a clock signal for the inoperative pod, insert the inoperative pod ribbon cable into the inoperative pod self test socket. (An alternative source for a clock signal is a known good UUT.)

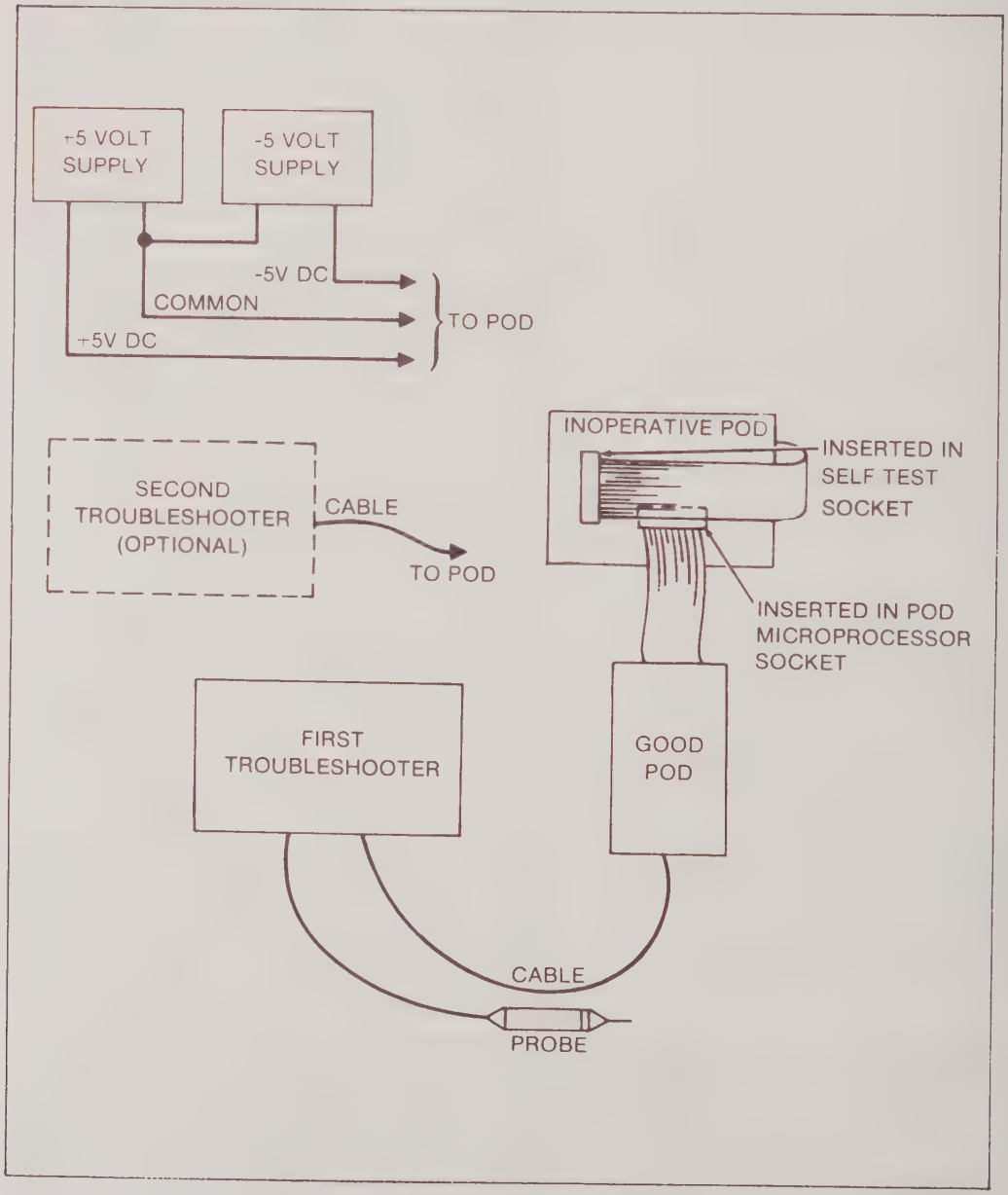


Figure 6-2. Troubleshooting an Inoperative Pod

5. If a second troubleshooter is available, connect the pod cable plug from the inoperative pod to the second troubleshooter to supply the inoperative pod with power. If a second troubleshooter is not available, connect a +5V dc (2A) power supply and a -5V dc (200 mA) power supply to the inoperative pod as shown in Figure 6-2. An easy place to make the power connections is at the pins on the connectors that connect the two pcbs together. Refer to the schematic diagrams in Section 8 for the proper pins for connection.

Troubleshoot the inoperative pod just as you would troubleshoot any other inoperative UUT. Refer to Table 6-4 for pod device addresses. Refer to Table 6-5 for bit definitions of selected devices and signals.

After the equipment is set up, begin by initializing the inoperative pod. If the pod is plugged into the second troubleshooter, press the BUS TEST key and then press the STOP key. The Bus Test will reset the inoperative pod; pressing the STOP key will prevent the troubleshooter from continuing to reset the pod. If the second troubleshooter is not available, use the first troubleshooter probe to pulse the inoperative pod Reset line.

After the inoperative pod is initialized, the first thing to try is a Bus Test. If you get a timeout message, disable the enable lines. If the Bus Test works now, check the input buffers. If a timeout still occurs, check the clock circuit.

The next thing to try is a ROM Test and a RAM Test. Refer to Table 6-4 for the ROM and RAM addresses, the expected ROM signature that should be obtained when the ROM Test is performed, and the expected ROM checksum that should be obtained when the Quick ROM Test is performed.

If the ROM Test and RAM Test are successful, the next thing to check is the data comunication with the troubleshooter. First check the path of incoming data. If a second troubleshooter is being used and has timed out, the data AB is probably being placed on the second troubleshooter data lines. The first troubleshooter can read the data lines of the second troubleshooter by performing a read operation at 1E00 (1E00 is the address of incoming data at U13 on the Processor pcb). The lower byte of data should be XXAB (X = any value). The most significant bit of the high byte should be the value of $\overline{\text{MAINSTAT}}$ with the rest of the bits indeterminate.

Table 6-4. Pod Device Addresses

DEVICE	LOWER ADDRESS	UPPER ADDRESS (IF ONE EXISTS)
RAM	0000	0FFE
ADDR LATCH	1800	—
DATA LATCH	1900	—
STATUS LATCH	1A00	—
CONTROL LATCH	1B00	—
TIMER	1C00	—
OUTPUT LATCH	1D00	—
DATA IN	1E00	—
DATA OUT	1F00	—
ROM * **	E000	FFFE

* For all troubleshooters, the data at FFFE equals the ROM signature if the ROM Test is performed at addresses E000-FFFC.

** For all troubleshooters, the ROM checksum obtained by performing the Quick ROM Test at addresses E000-FFFC equals 0002.

Table 6-5. Bit Definitions for Selected Pod Addresses

ADDRESS*	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
(DATA TO/FROM 9000)								
1E01	$\overline{\text{MAINSTAT}}$	—	—	—	—	—	—	—
1E00	POD7	POD6	POD5	POD4	POD3	POD2	POD1	POD0
1F00	POD7	POD6	POD5	POD4	POD3	POD2	POD1	POD0
(OUTPUT PORT)								
1D01	$\overline{\text{PODSTAT}}$	9KOUT	DATA/$\overline{\text{ADD}}$	RUNREQ	ENINTR	ENRQ1	ENRQ0	ENRDY
1D00	FINT	WTCTL	SPECIAL	FGT1	FGT0	FQS1	FQS0	FLOCK
(STATUS LATCH)								
1A01	A19	A18	A17	A16	$\overline{\text{BHE}}$	$\overline{\text{S2}}$:M/$\overline{\text{IO}}$	$\overline{\text{S1}}$:DT/$\overline{\text{R}}$	$\overline{\text{S0}}$:$\overline{\text{DEN}}$
1A00	STS	SLFTST	PWRFL	HOLD1	RESET	INTR	NMI	HOLD0
(CONTROL LATCH)								
1B01	S7	S6	S5	S4	S3	0	$\overline{\text{TEST}}$	MN/$\overline{\text{MX}}$
1B00	DALE	$\overline{\text{RD}}$	0	GT1	GT0	QS1:INTA	QS0:ALE	$\overline{\text{LOCK}}$

*If address is odd, a byte access prefix must be specified (such as 800 1E01).

If a second troubleshooter is not being used, you may check the incoming data path by performing a looping read operation at 1E00. Set the probe stimulus to toggle high and low pulses; then apply the stimulus pulses to the data lines and check whether all bits are readable high and low.

If you are using a second troubleshooter, checking the path of outgoing data is not as straightforward as previously described for incoming data. If you are using a second troubleshooter, disconnect the inoperative pod and connect a good pod to the second troubleshooter. Do a complete operation that does not time out; then press the STOP key. This will leave the troubleshooter data lines in a tristate condition. Without disconnecting power, disconnect the good pod from the second troubleshooter and reconnect the inoperative pod. Do not press any keys on the second troubleshooter. Using the first troubleshooter probe in the probe stimulus mode, pulse the Reset line on the inoperative pod to initialize the inoperative pod.

On the first troubleshooter, perform the operation Write @ 1D00 to set Bit 14 high; this should turn on the data output latch. Then write a value to 1F00 and check the data lines with the probe to see if the expected value is present.

Next check the output port to make sure all signals are writable. Make sure a pod reset resets all output port lines low.

Next try writing to the timer. Use the probe in the free-run mode. If RUNREQ is low, the UUTON should have a short pulse. If RUNREQ is high, RUNUUT should become active after the timer fires, and stay on until the pod is reset. This is the Run UUT mode. While in Run UUT mode, you should not be able to read the ROM or RAM of the

inoperative pod, but instead you should be communicating with the inoperative pod UUT or the self test socket.

If all these tests are good, you should no longer have an inoperative pod, but you may still have a defective pod. If the pod is still defective, refer to the previous section titled Troubleshooting a Defective Pod.

If you are using a second troubleshooter, there is an easy way to test the microprocessor that was removed from the inoperative pod. Place the first troubleshooter in the Run UUT mode and try a pod self test on the inoperative pod. If the pod self test is successful, remove the good pod from the inoperative pod, reinsert the original microprocessor, and try the pod self test again. If the pod self test is not successful, replace the microprocessor.

6-10. EXTENDED TROUBLESHOOTING PROCEDURES

The troubleshooting procedures provided in this section supplement the circuit checks performed on the pod during the pod self test; these procedures are appropriate for use with a pod that passes the pod self test but does not appear to function normally when used with a troubleshooter and a good UUT. If a pod fails self test, it would be better to begin troubleshooting with the procedure provided in the section titled Troubleshooting a Defective Pod.

Potential problems that may exist in a pod that passes the pod self test may be divided into two categories. The two categories are partially checked circuits and timing and noise problems.

6-11. Partially Checked Circuits

The most obvious partially checked circuits, and the easiest to check, are possible open lines on the UUT cable. The circuits are usually checked through the hybrid protection circuits and back through the latches, but the cable lines themselves are not checked. Refer to Table 6-6 for a list of the partially checked lines. These lines can be checked with an ohmmeter, but if they are intermittent, a better way is to check them with the troubleshooter probe or an oscilloscope while the pod is exibiting its symptoms. Check at the pod end for the status lines (inputs), and check at the UUT end for control lines (outputs).

During the pod self test, the pod self test socket forces the pod into the minimum mode, so some of the maximum mode circuits are only partially checked. These include the RQ/GT circuits, the QS0 and QS1 circuits, and any signal that is different in the maximum mode from the minimum mode.

Table 6-6. Pod Ribbon Cable Lines Partially Checked in Pod Self Test

STATUS LINES		CONTROL LINES	
SIGNAL NAME	PIN NO.	SIGNAL NAME	PIN NO.
NMI	17	A16	38
INTR	18	A17	37
RESET	21	A18	36
READY	22	A19	35
$\overline{\text{TEST}}$	23	$\overline{\text{BHE}}$	34
HOLD	31	HLDA	30
MN/$\overline{\text{MX}}$	33	$\overline{\text{WR}}$	29
		M/$\overline{\text{IO}}$	28
		DT/$\overline{\text{R}}$	27
		$\overline{\text{DEN}}$	26
		$\overline{\text{INTA}}$	24

The circuits which generate the transparent reads to the UUT when the pod is not in the UUTON mode are also partially checked. These circuits could be malfunctioning between cycles and causing problems with the UUT.

Another circuit which is partially checked is the tristate and wait state logic. If the UUT uses DMA cycles or wait states, these could be the problem.

6-12. Timing and Noise Problems

Timing or noise problems are usually caused by components that are still functioning, but are not functioning within the component specifications. The best way to check this problem is to look at suspected signals using an oscilloscope synchronized to either address or data. Look for slow rise times or signals driven to marginal levels. If the part is too slow, it might fail on the UUT, but pass the pod self test because the clock signal to the pod is slightly slower.

If a part has marginal drive capabilities, the added noise of a UUT environment might cause it to fail. Be sure to note that inputs can malfunction (they may leak perhaps) and put too much load on an output causing either low levels, slow rise times, or both.

6-13. DISASSEMBLY

To gain access to the two pcb assemblies in the pod, perform the following steps:

1. Remove the pod ribbon cable plug from the self test socket.

2. Turn the pod over on its top (with the large pod decal facing up). Remove the four phillips screws that hold the case halves together and remove the top and bottom case halves. Place the pcb assemblies so that the self test socket (on the processor pcb assembly) is facing up.

3. Remove the four phillips screws that connect the heat sink to the processor pcb assembly and remove the heat sink.

NOTE

The heat sink is not needed for heat dissipation unless the pod is fully assembled. If the two pcb assemblies are removed from the top and bottom cases, the heat sink may be removed during pod operation and troubleshooting.

4. On the corner opposite the self test socket thumbwheel, remove the single phillips screw that retains the shield surrounding the pcb assemblies. (A standoff and washer will come off with the screw.) Remove the shield.

NOTE

When the shield and the heat sink are removed, all the components are exposed. It should not be necessary to separate the two pcb assemblies while troubleshooting except to replace components.

5. To separate the two pcb assemblies, turn the pcb assemblies over so that the self test socket is facing down. Remove the four phillips screws at the corners of the pcb assemblies and carefully pull the boards apart at the two connectors along the sides.

Section 7
List of Replaceable Parts

7-1. INTRODUCTION

This section contains an illustrated parts list for the instrument. Components are listed alphanumerically by assembly.

Parts lists include the following information:

1. Reference Designation.

2. Description of Each Part.

3. Fluke Stock Number.

4. Federal Supply Code for Manufacturers (see the 9000 Series Troubleshooter Service Manual for Code-to-Name list).

5. Manufacturer's Part Number.

6. Total Quantity of Components Per Assembly.

7. Recommended quantity: This entry indicates the recommended number of spare parts necessary to support one to five instruments for a period of 2 years. This list presumes an availability of common electronic parts at the maintenance site. For maintenance for 1 year or more at an isolated site, it is recommended that at least one of each assembly in the instrument be stocked.

7-2. HOW TO OBTAIN PARTS

Components may be ordered directly from the manufacturer by using the manufacturer's part number, or from the John Fluke Mfg. Co., Inc. or an authorized representative by using the Fluke Stock Number.

In the event the part ordered has been replaced by a new or improved part, the replacement will be accompanied by an explanatory note and installation instructions if necessary.

To ensure prompt and efficient handling of your order, include the following information.

1. Quantity.

2. Fluke Stock Number.

3. Description.

4. Reference Designation.

5. Printed Circuit Board Part Number and Revision Letter.

6. Instrument Model and Serial Number.

A Recommended Spare Parts Kit for your basic instrument is available from the factory. This kit contains those items listed in the REC QTY column for the parts lists in the quantities recommended.

Parts price information is available from the John Fluke Mfg. Co., Inc. or its representative. Prices are also available in a Fluke Replacement Parts Catalog, which is available upon request.

CAUTION

Indicated devices are subject to damage by static discharge.

7-3. MANUAL CHANGE AND BACKDATING INFORMATION

Table 7-4 contains information necessary to backdate the manual to conform with earlier pcb configurations. To identify the configuration of the pcbs used in your instrument, refer to the revision letter on the component side of each pcb assembly.

As changes and improvements are made to the instrument, they are identified by incrementing the revision letter marked on the affected pcb assembly. These changes are documented on a supplemental change/errata sheet which, when applicable, is inserted at the front of the manual.

To backdate this manual to conform with an earlier assembly revision level, perform the changes indicated in Table 7-4. There are no backdating changes at this printing. All pcb assemblies are documented at their original revision level.

Table 7-1. 9000A-8086 Final Assembly

REF DES	DESCRIPTION	FLUKE STOCK NO.	MFG SPLY CODE	MFG PART NO.	TOT QTY	REC QTY	NOTE
	⊗ FINAL ASSEMBLY, 9000A-8086 POD FIGURE 7-1 (9000A-8086-5001)						
A27⊗	PROCESSOR PCB ASSEMBLY	647990	89536	647990	1		
A28⊗	INTERFACE PCB ASSEMBLY	648014	89536	648014	1		
H1	SCREW, CAP, 4-40 X 3/8	571935	89536	571935	4		
H2	SCREW, PHP, 4-40 X 1/4	185918	89536	185918	4		
H3	SCREW, PHP, 4-40 X 5/8	145813	89536	145813	1		
H4	SCREW, RHP, 4-40 X 3/4	115063	89536	115063	4		
H5	WASHER, #4 LOCK	110403	89536	110403	1		
MP1	ACTUATOR	582916	89536	582916	1		
MP2	DECAL, POD	659144	89536	659144	1		
MP3	DECAL, SPECIFICATION	659151	89536	659151	1		
MP4	DECAL, WARNING	659805	89536	659805	1		
MP5	HEAT SINK	654483	89536	654483	1		
MP6	HEAT SINK CLIP	607671	89536	607671	3		
MP7	HEAT SINK CLIP	607655	89536	607655	4		
MP8	LABEL, STATIC CAUTION	605808	89536	605808	1		
MP9	LABEL, UUT CAUTION	634030	89536	634030	1		
MP10	SHELL, BOTTOM	648881	89536	648881	1		
MP11	SHELL, TOP	648899	89536	648899	1		
MP12	SHIELD	659797	89536	659797	1		
MP13	SPACER, HEX	187575	89536	187575	1		
TM1	INSTRUCTION MANUAL, 9000A-8086	649426	89536	649426			
U1⊗	IC, 16-BIT MICROPROCESSOR	647180	34649	8086-1	1	1	
U2⊗	IC, ROM, PROGRAMMED (violet dot)	653105	89536	653105	1	1	
U4⊗	IC, ROM, PROGRAMMED (blue dot)	653097	89536	653097	1	1	
U15	IC, PAL10L8, PROGRAMMED (red dot)	540450	89536	540450	1	1	
U23	IC, PAL16R8, PROGRAMMED (green dot)	540435	89536	540435	1	1	
U24	IC, PAL16L8, PROGRAMMED (yellow dot)	540427	89536	540427	1	1	
U25	IC, PAL16L8, PROGRAMMED (orange dot)	540443	89536	540443	1	1	
U41	IC, PAL16R4, PROGRAMMED (white dot)	583252	89536	583252	1	1	
W1	CABLE, POD	607184	89536	607184	1		
W2	CABLE, UUT	680603	89536	680603	1		
	RECOMMENDED SPARE PARTS KIT (9000A-8086)	661157	89536	661157			
	ACCESSORY KIT	607622	89536	607622	1		
	40-PIN DIP SOCKET SAVER	614297	89536	614297			
	PROGRAMMED HEADER	659839	89536	659839	1	1	

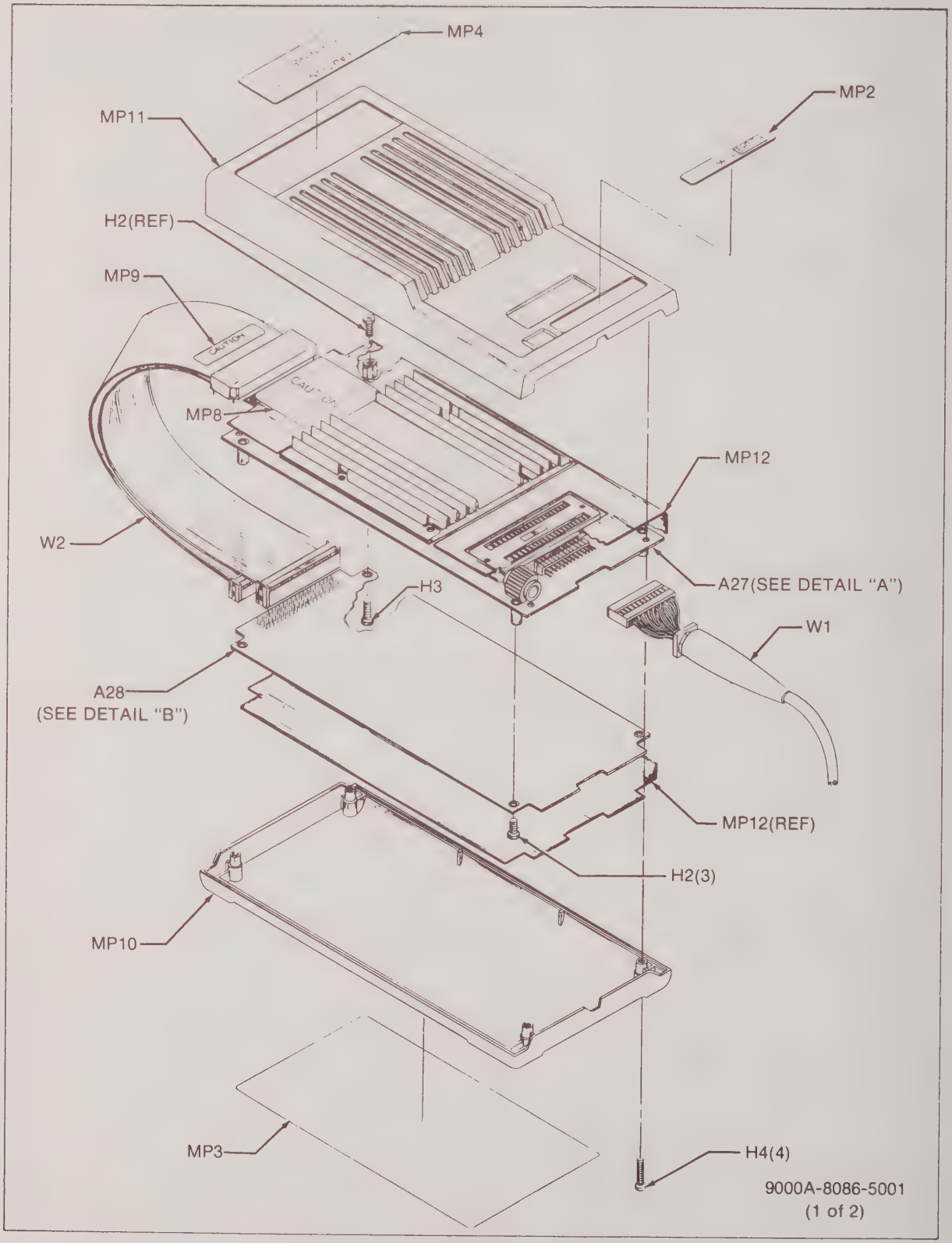
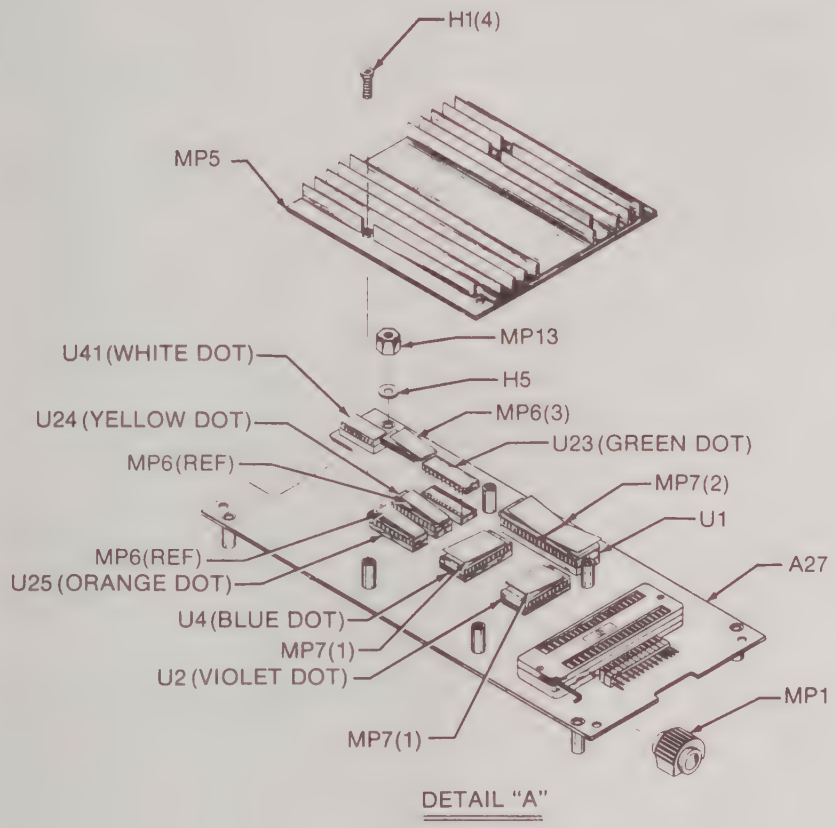
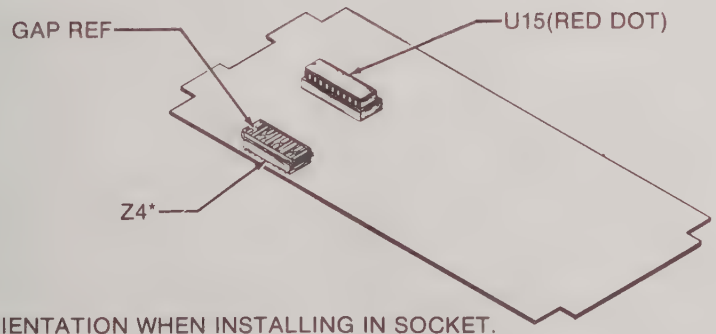


Figure 7-1. 9000A-8086 Final Assembly



DETAIL "A"



*NOTE ORIENTATION WHEN INSTALLING IN SOCKET.

DETAIL "B"

9000A-8086-5001
(2 of 2)

Figure 7-1. 9000A-8086 Final Assembly (cont)

Table 7-2. A27 Processor PCB Assembly

REF DES	DESCRIPTION	FLUKE STOCK NO.	MFG SPLY CODE	MFG PART NO.	TOT QTY	REC QTY	NOTE
A27Ø	PROCESSOR PCB ASSEMBLY FIGURE 7-2 (9000A-8086-4071)	647990	89536	647990	REF		
C1-C14	CAP, CER 0.22 UF +/-20%, 50V	519157	51406	RPE111Z5U224M50V	14		
J1	CONNECTOR, POST, RIGHT ANGLE	512590	89536	512590	1		
J2	CONNECTOR, SOCKET, 40-PIN, ZIF	585133	89536	585133	1		
MP1	SPACER, STANDOFF, BRASS (NOT SHOWN)	442913	89536	442913	4		
MP2	STANDOFF, SPACER SWAGE (NOT SHOWN)	380329	89536	380329	4		
P1, P2	CONNECTOR	267500	00779	87022-1	100		
R1	RES, DEP. CAR, 39 +/-5%, 1/4W	340836	80031	CR251-4-5P39E	1		
R3	RES, DEP. CAR, 100 +/-5%, 1/4W	348771	80031	CR251-4-5P100E	1		
R4-R7	RES, DEP. CAR, 4.7K +/-5%, 1/4W	348821	80031	CR251-4-5P4K7	4		
TP1-TP4	TERMINAL, TEST POINT	179283	89536	179283	4		
U1Ø	IC, 16-BIT MICROPROCESSOR		see	Table 7-1	REF		
U2Ø	IC, ROM, PROGRAMMED (violet dot)		see	Table 7-1	REF		
U3Ø	IC, CMOS, 2K X 8-BIT STATIC RAM	647222	51157	HM6116P-3	2	1	
U4Ø	IC, ROM, PROGRAMMED (blue dot)		see	Table 7-1	REF		
U5Ø	IC, CMOS, 2K X 8-BIT STATIC RAM	647222	51157	HM6116P-3	REF		
U6	IC, LSTTL, OCTAL + EDGE TRG F/F W/TRIST	473223	01295	SN74LS374N	1	1	
U7	IC, LSTTL, OCTAL BUFFER/LINE DRIVER	634105	04713	SN74LS541N	1	1	
U8, U9Ø	IC, CMOS, OCTAL BUS TRANSCEIVER	535906	36665	74SC245AC	2	1	
U10, U11	IC, LSTTL, OCTAL D TRNSPRNT LTCHS TRI	504514	01295	SN74LS373N	4	1	
U12, U13	IC, LSTTL, 3-8 BIT DECODER W/ENABLE	407585	01295	SN74LS138N	2	1	
U14, U15	IC, LSTTL, OCTAL D TRNSPRNT LTCHS TRI	504514	01295	SN74LS373N	REF		
U16	IC, LSTTL, QUAD 2-INPUT NAND GATE	393033	01295	SN74LS00N	2	1	
U17	IC, FTTL, HEX INVERTER	634444	07235	74F04PC	1	1	
U18	IC, LSTTL, QUAD 2-INPUT NAND BFR OPN COL	524736	01295	SN74LS38N	1	1	
U19	IC, LSTTL, TRIPLE 3-INPUT NOR GATE	393090	01295	SN74LS27N	1	1	
U20	IC, LSTTL, DUAL +EDGE TRG FF W/PRST&CLR	393124	01295	SN74LS74N	3	1	
U21	IC, LSTTL, QUAD BFR GTS @ TRI-ST NOT	472746	01295	SN74LS125AN	2	1	
U22	IC, LSTTL, QUAD 2-INPUT OR GATE	393108	01295	SN74LS32N	2	1	
U23	IC, PAL16R8, PROGRAMMED (green dot)		see	Table 7-1	REF		
U24	IC, PAL16L8, PROGRAMMED (yellow dot)		see	Table 7-1	REF		
U25	IC, PAL16L8, PROGRAMMED (orange dot)		see	Table 7-1	REF		
U26	IC, FTTL, QUAD 2-INPUT MULTIPLEXER	647156	07263	74F257PC	1	1	
U27	IC, FTTL, DUAL D F/F, POS EDGE TRIG	659508	07263	74F74PC	1	1	
U28	IC, LSTTL, SYNCHRNS 4-BIT UP/DWN CNTR	393231	01295	SN74LS193N	1	1	
U29	IC, LSTTL, DUAL +EDGE TRG FF W/PRST&CLR	393124	01295	SN74LS74N	REF		
U30	IC, LSTTL, QUAD BFR GTS @ TRI-ST NOT	472746	01295	SN74LS125AN	REF		
U31	IC, LSTTL, QUAD 2-INPUT OR GATE	393108	01295	SN74LS32N	REF		
U32, U33	IC, LSTTL, OCTAL + EDGE TRG D F/F W/CLR	454892	01295	SN74LS273N	2	1	
U34	IC, LSTTL, DUAL +EDGE TRG FF W/PRST&CLR	393124	01295	SN74LS74N	REF		
U35	IC, LSTTL, TRIPLE 3-INPUT NAND GATE	393074	01295	SN74LS10N	1	1	
U36	IC, LSTTL, QUAD 2-INPUT AND GATE	393066	01295	SN74LS08N	2	1	
U37	IC, FTTL, QUAD 2-INPUT AND GATE	650523	07263	74F08PC	1	1	
U38	IC, LSTTL, QUAD 2-INPUT AND GATE	393066	01295	SN74LS08N	REF		
U39	IC, LSTTL, QUAD 2-INPUT NOR, BIPOLAR	393041	01295	SN74LS02N	1	1	
U40	IC, LSTTL, QUAD 2-INPUT NAND GATE	393033	01295	SN74LS00N	REF		

Table 7-2. A27 Processor PCB Assembly (cont)

REF DES	DESCRIPTION	FLUKE STOCK NO.	MFG SPLY CODE	MFG PART NO.	TOT QTY	REC QTY	NOTE
U41	IC, PAL16R4, PROGRAMMED (white dot)		see	Table 7-1	REF		
W1	WIRE, JUMPER, #22 AWG	529701	89536	529701	1		
XU1	SOCKET, IC, 40-PIN	429282	09922	DILB40P-108	1		
XU2-XU5	SOCKET, IC, 24-PIN	376236	91506	324-AG39D	4		
XU23-25	SOCKET, IC, 20-PIN	454421	01295	C932002	4		
XU41	SOCKET, IC, 20-PIN	454421	01295	C932002	REF		
XY1	SPACER (W/Y1)	441865	32559	814-060	1		
Y1	CRYSTAL, OSCILLATOR, 6.5 MHZ, +/-1%	586933	89536	586933	1	1	
Z1	RESISTOR NETWORK, 4.7K	412916	89536	412916	1	1	

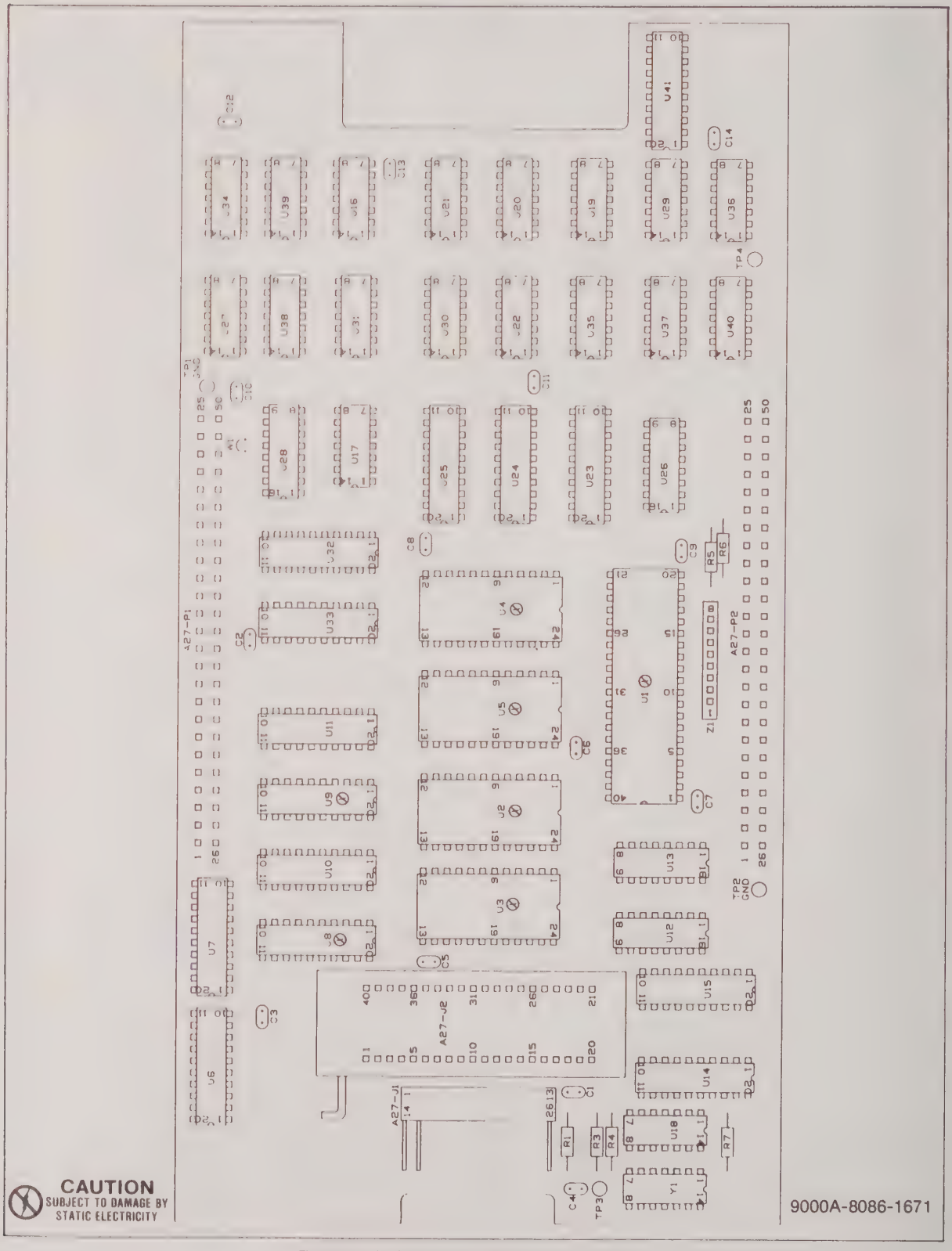


Figure 7-2. A27 Processor PCB Assembly

Table 7-3. A28 Interface PCB Assembly

REF DES	DESCRIPTION	FLUKE STOCK NO.	MFG SPLY CODE	MFG PART NO.	TOT QTY	REC QTY	NOTE
A28Ø	INTERFACE PCB ASSEMBLY FIGURE 7-3 (9000A-8086-4072)	648014	89536	648014	REF		
A1-A5	PROTECTION NETWORK, HYBRID, 700 OHMS	582189	89536	582189	5	1	
C1, C2	CAP, TA, 10 UF +/-20%, 15V	193623	56289	196D106X0015KA1	4		
C3	CAP, CER, 0.01 UF +/-20%, 100V	407361	72982	8121-A100-W5R-103M	2		
C4-C8	CAP, CER, 0.22 UF +/-20%, 50V	519157	51406	RPE111Z5U224M50V	14		
C9	CAP, CER, 0.01 UF +/-20%, 100V	407361	72982	8121-A100-W5R-103M	REF		
C10-C17	CAP, CER, 0.22 UF +/-20%, 50V	519157	51406	RPE111Z5U224M50V	REF		
C18, C19	CAP, TA, 10 UF +/-20%, 15V	193623	56289	196D106X0015KA1	REF		
C20	CAP, CER, 0.22 UF +/-20%, 50V	519157	51406	RPE111Z5U224M50V	REF		
CR1, CR2	DIODE, SI, SMALL SIGNAL	313247	28484	HP5082-6264	2	1	
J1, J2	CONNECTOR, 50-POS	649848	00779	86396-5	2		
J3A, J3B	CONNECTOR, POST	267500	00779	87022-1	80		
R1	RES, DEP. CAR, 820 +/-5%, 1/4W	442327	80031	CR251-4-5P820E	1		
R2	RES, DEP. CAR, 3K +/-5%, 1/4W	441527	80031	CR251-4-5P3K	1		
R3, R4	RES, DEP. CAR, 200 +/-5%, 1/4W	441451	80031	CR251-4-5P200E	2		
R5	RES, DEP. CAR, 2.2K +/-5%, 1/4W	343400	80031	CR251-4-5P2K2	1		
R6	RES, DEP. CAR, 1K +/-5%, 1/4W	343426	80031	CR251-4-45P1K	1		
R7	RES, MTL. FILM, 22.6K +/-1%, 1/8W	288431	91637	CMF552262F	1		
R8	RES, MTL. FILM, 12.1K +/-1%, 1/8W	234997	91637	CMF551212F	1		
R9	RES, COMP, 510K +/-5%, 1/4W	275685	01121	CB5145	1		
R10	RES, DEP. CAR, 47K +/-5%, 1/4W	348896	80031	CR251-4-5P47K	1		
TP1, TP2	TERMINAL, TEST POINT	179283	89536	179283	2		
U1, U2Ø	IC, CMOS, OCTAL D F/F W/3-STATE	585364	36665	74SC374A	8	2	
U3	IC, SLSSTL, OCTAL BUS XCVR W/3-STATE	647214	01295	SN74ALS245N	2	1	
U4	IC, TTL, OCTAL BUFFER/LINE DRIVER	634105	04713	SN74LS541N	2	1	
U5, U6Ø	IC, CMOS, OCTAL D F/F W/3-STATE	585364	36665	74SC374A	REF		
U7	IC, SLSSTL, OCTAL BUS XCVR W/3-STATE	647214	01295	SN74ALS245N	REF		
U8	IC, TTL, OCTAL BUFFER/LINE DRIVER	634105	04713	SN74LS541N	REF		
U9, U10Ø	IC, CMOS, OCTAL D F/F W/3-STATE	585364	36665	74SC374A	REF		
U11, U12	IC, FTTL, QUAD, 2-INPUT MULTIPLEXER	647156	01295	74F257PC	2	1	
U13Ø	IC, CMOS, OCTAL D F/F W/3-STATE	585364	36665	74SC374A	REF		
U14	IC,DGTL,LSTTL,QUAD BFR GTS @ TRI-ST NOT	472746	01295	AN74LS125AN	1	1	
U15	IC, PAL10L8, PROGRAMMED (red dot)		see	Table 7-1	REF		
U16Ø	IC, CMOS, OCTAL D F/F W/3-STATE	585364	36665	74SC374A	REF		
U17	IC, LSTTL, QUAD 2-INPUT OR GATE	393108	01295	SN74LS32N	1	1	
U18	IC,LSTTL,DUAL-EDG TRG JK FF W/PRST & CL	414029	01295	SN74LS112N	1	1	
U19	IC, FTTL, DUAL D F/F POS EDGE TRIG	659508	07263	74F74PC	1	1	
U20, U21	IC,LSTTL, DUAL + EDGE TRG D FF W/PRST&CL	393124	01295	SN74LS74N	2	1	
U22	IC, LSTTL, TRIPLE 3-INPUT NAND GATE	393074	01295	SN74LS10N	1	1	
U23	IC, DGTL, LSTTL, QUAD 2-INPUT	393041	01295	SN74LS02N	1	1	
U24	IC, PROTECTION	585992	89536	585992	1	1	
U25	IC, LINEAR, VOLTAGE COMPARATOR	352195	01295	SN72311P	1	1	
VR1	IC, LINEAR, 1.22V BNDGAP REF (SELECTED)	452771	89536	452771	1	1	
W8, W10	WIRE, JUMPER	529776	89536	529776	2		
XU3, XU4	SOCKET, IC, 20-PIN	454421	01295	C932002	5		
XU7, XU8	SOCKET, IC, 20-PIN	454421	01295	C932002	REF		
XU15	SOCKET, IC, 20-PIN	454421	01295	C932002	REF		
XU24	SOCKET, IC, 18-PIN	418228	91506	318-AG39D	1		

Table 7-3. A28 Interface PCB Assembly (cont)

REF DES	DESCRIPTION	FLUKE STOCK NO.	MFG SPLY CODE	MFG PART NO.	TOT QTY	REC QTY	NOTE
XVR1	INSULATOR W/VR1	175125	89536	175125	2		
XZ4	SOCKET, 16-PIN	276535	91506	316-AG39D	1		
Z1	RESISTOR NETWORK	583476	89536	583476	1	1	
Z2	RESISTOR NETWORK, 4.7K +/-2%, 1/8W	484063	89536	484063	1	1	
Z3	RESISTOR NETWORK, 4.7K +/-2%, 1/8W	412916	89536	412916	1	1	

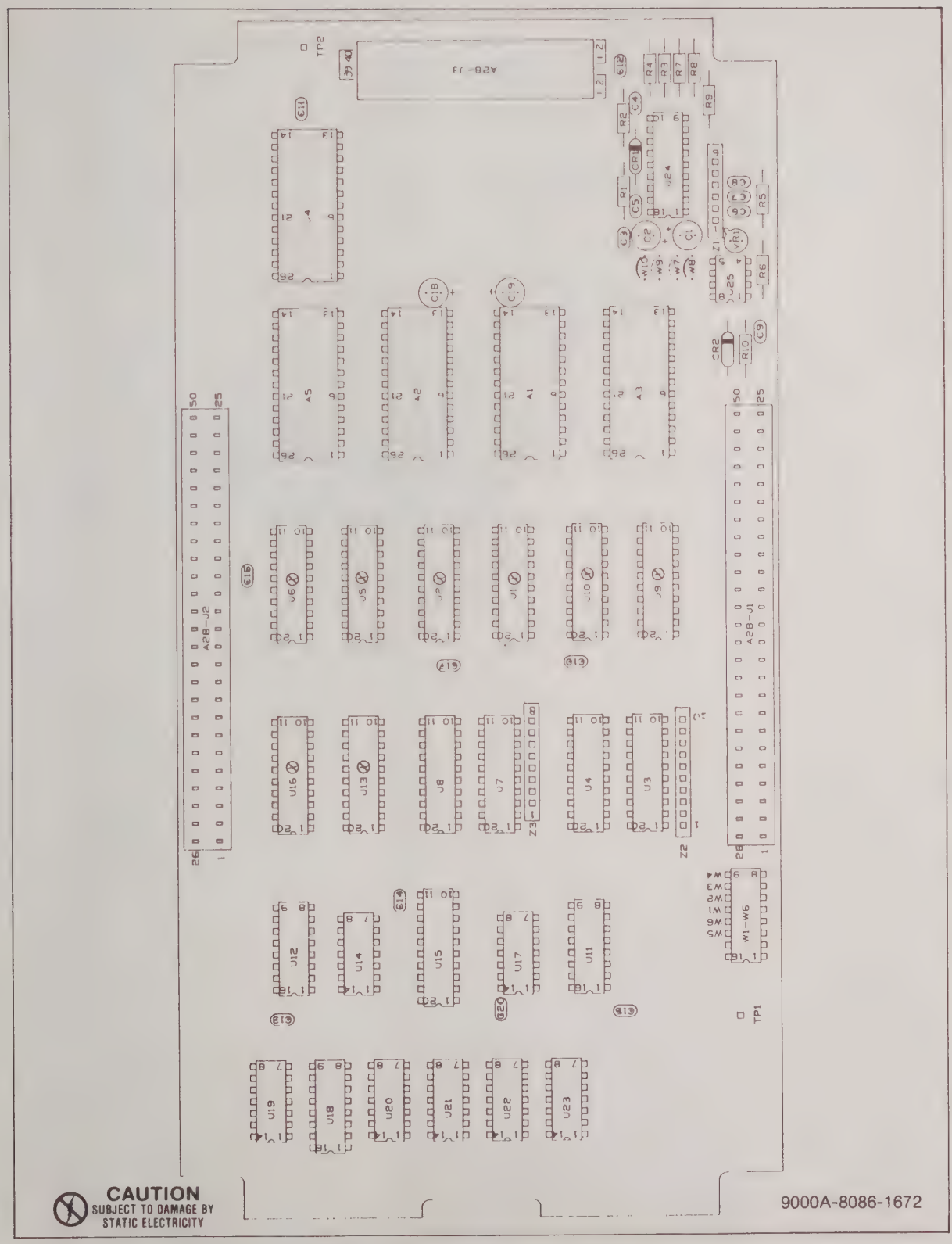


Figure 7-3. A28 Interface PCB Assembly

Table 7-4. Manual Status and Backdating Information

Ref Or Option No.	Assembly Name	Fluke Part No.	* To adapt manual to earlier rev configurations perform changes in desending order (by no.), ending with change under desired rev letter																					
			–	A	B	C	D	E	F	G	H	J	K	L	M	N	P							
A27	Processor PCB Assembly	647990	●	●	X																			
A28	Interface PCB Assembly	648014	●	●	X																			

* X = The PCB revision levels documented in this manual.
● = These revision letters were never used in the instrument.
– = No revision letter on the PCB.

Section 8
Schematic Diagrams

TABLE OF CONTENTS

FIGURE	TITLE	PAGE
8-1.	Schematic Diagram of U15 (on A28 Interface PCB)	8-3
8-2.	Schematic Diagram of U24 (on A27 Processor PCB)	8-4
8-3.	Schematic Diagram of U25 (on A27 Processor PCB)	8-5
8-4.	A27 Processor PCB Assembly	8-6
8-5.	A28 Interface PCB Assembly	8-9

NOTES;

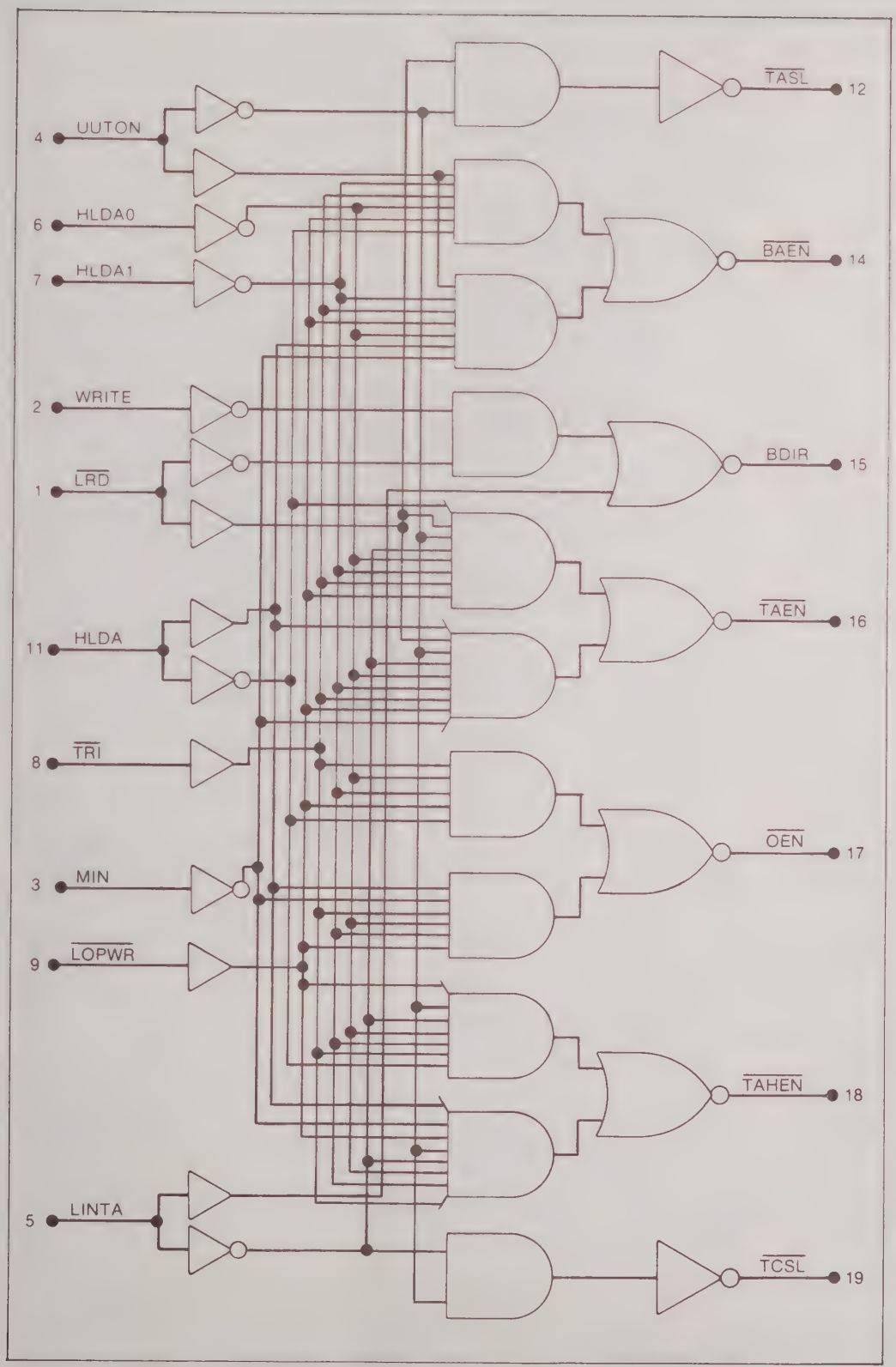


Figure 8-1. Schematic Diagram of U15 (on A28 Interface PCB)

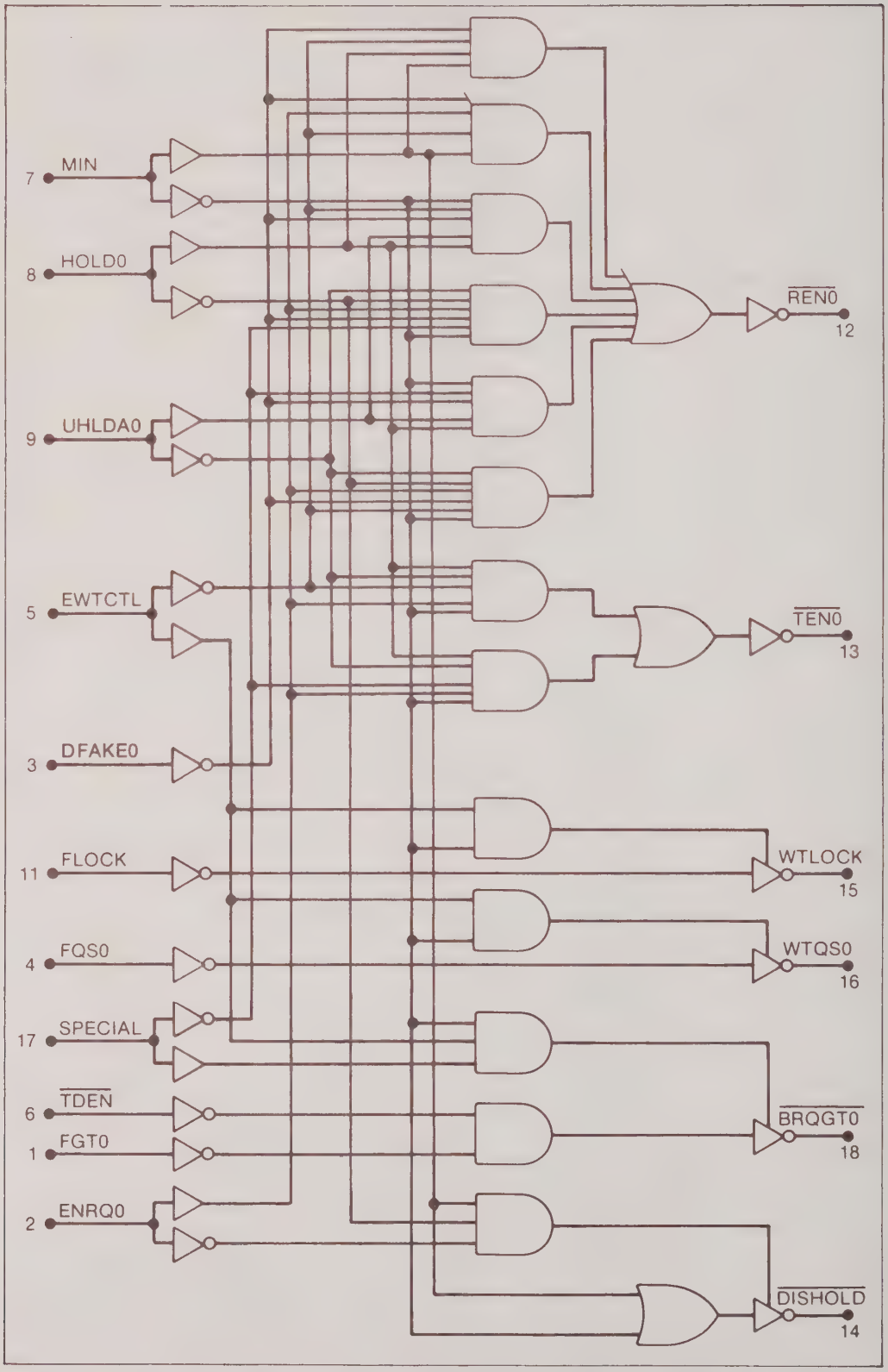


Figure 8-2. Schematic Diagram of U24 (on A27 Processor PCB)

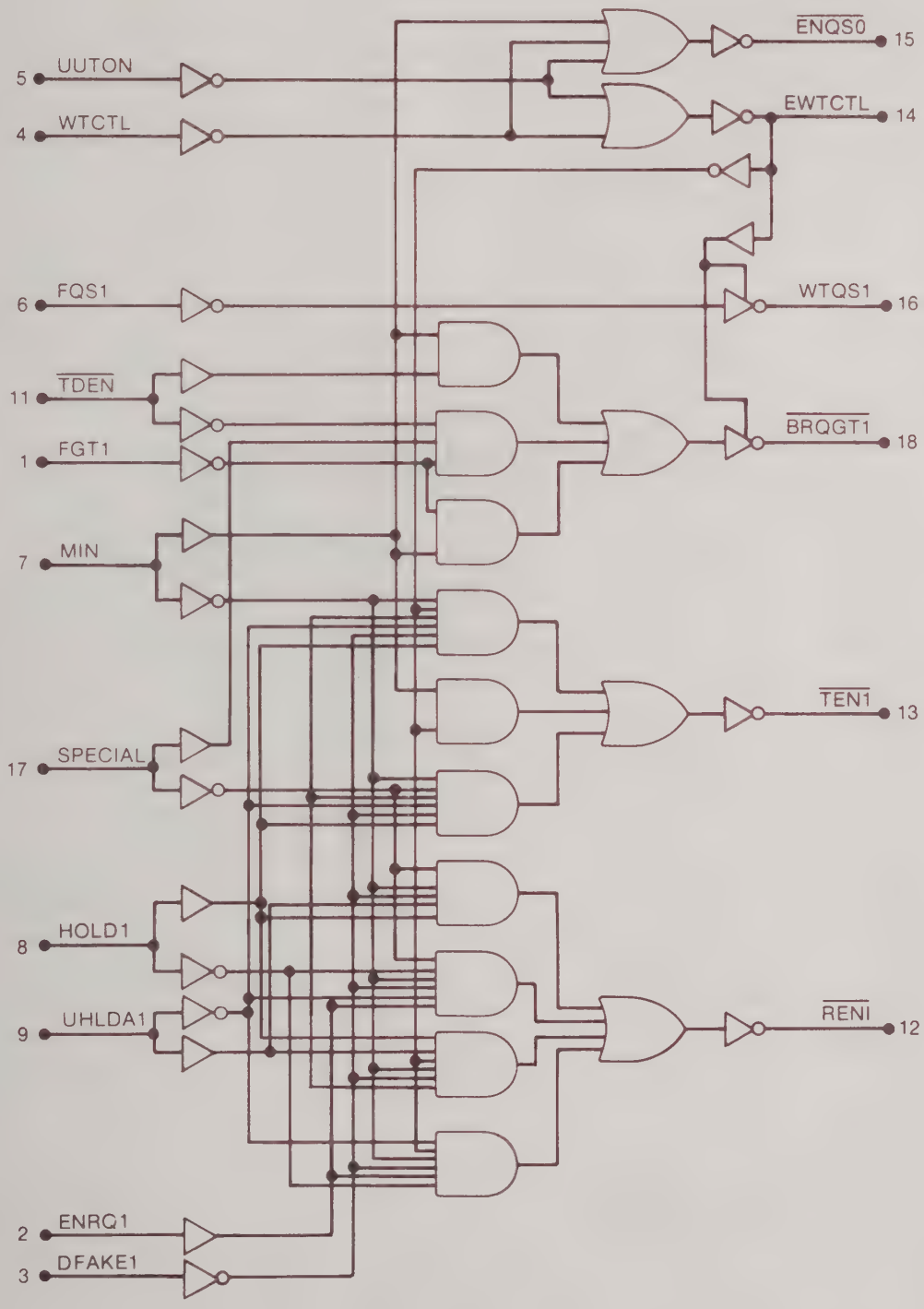
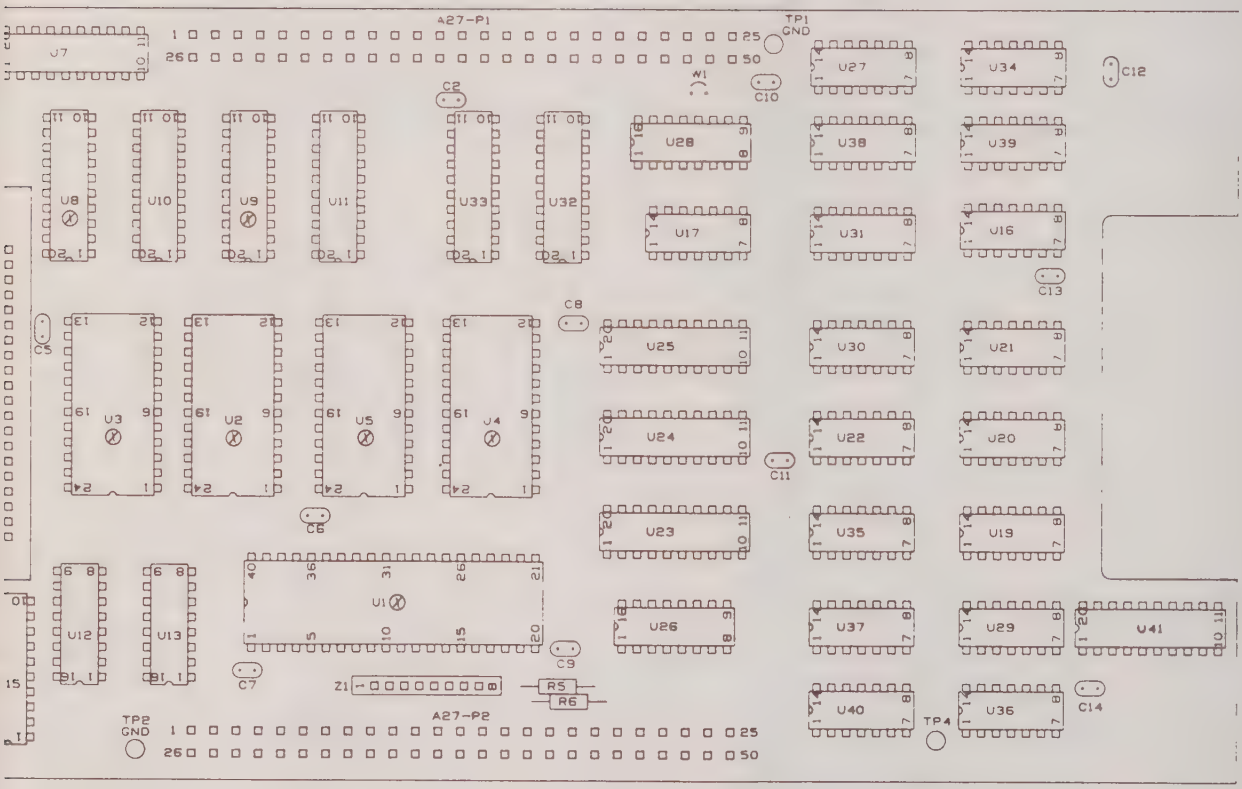

5 UUTON
4 WTCTL
6 FQS1
11 TDEN
1 FGT1
7 MIN
17 SPECIAL
8 HOLD1
9 UHLDA1
2 ENRQ1
3 DFAKE1
ENQS0 15
EWTCTL 14
WTQS1 16
BRQGT1 18
TEN1 13
RENI 12

Figure 8-3. Schematic Diagram of U25
(on A27 Processor PCB)

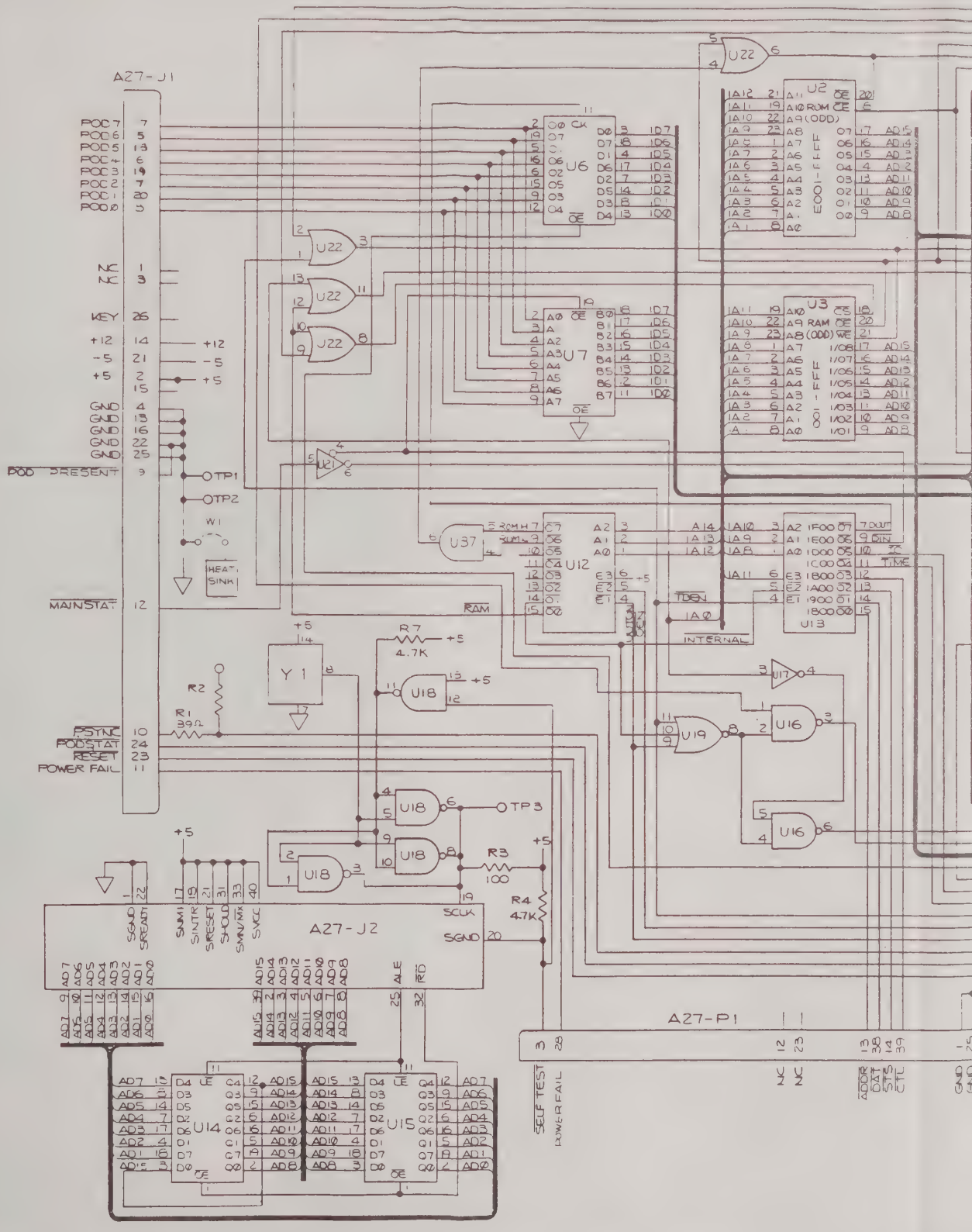
U5
A27-J1
14 1
2613
C4
TP3
R1
R3
R4
C1
Y1
U18
U14
R7



CAUTION
SUBJECT TO DAMAGE BY STATIC ELECTRICITY

9000A-8086-1671

Figure 8-4. A27 Processor PCB Assembly


A27-J1
A27-J2
A27-P1
POD7
POD6
POD5
POD4
POD3
POD2
POD1
POD0
KEY
+12
-5
+5
GND
POD PRESENT
TP1
TP2
W1
HEAT SINK
MAINSTAT
PSYNC
PODSTAT
RESET
POWER FAIL
R1 39Ω
R2
Y1
R7 4.7K
R3 100
R4 4.7K
TP3
U2
U3
U6
U7
U12
U13
U14
U15
U16
U17
U18
U19
U21
U22
U37
RAM
INTERNAL
TDEN
SELF TEST
LOWER FAIL

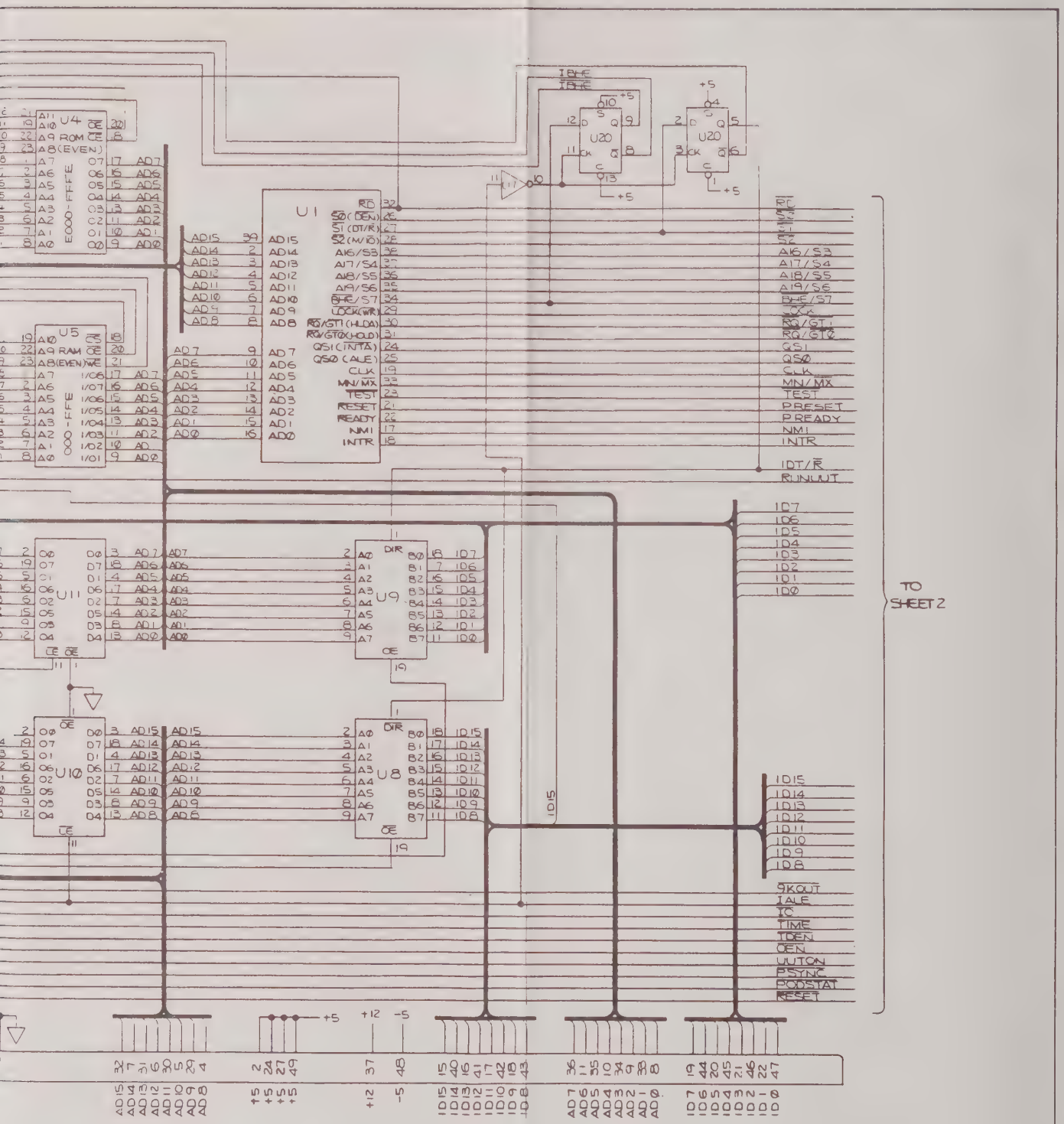
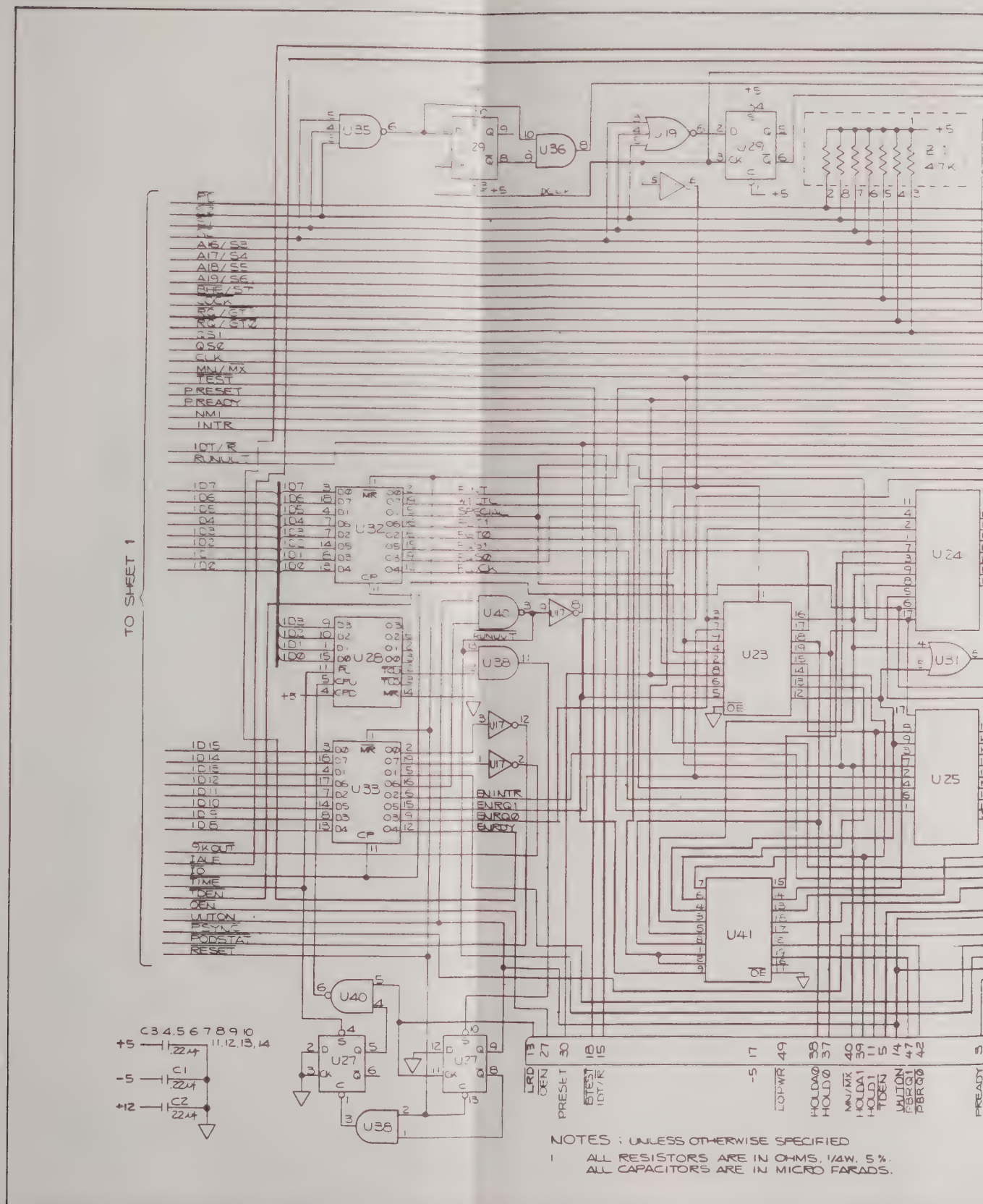
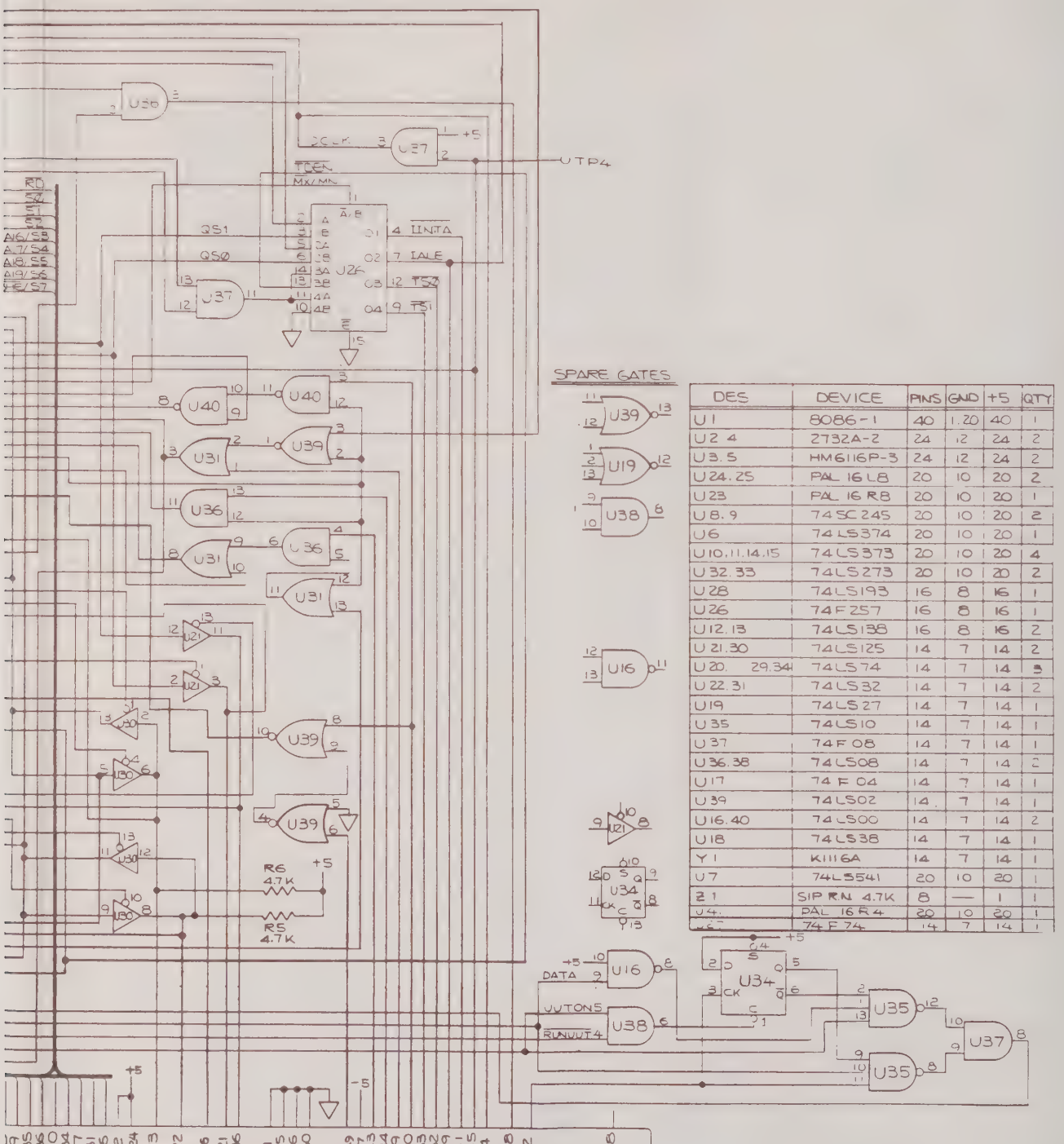


Figure 8-4. A27 Processor PCB Assembly (cont)



NOTES : UNLESS OTHERWISE SPECIFIED

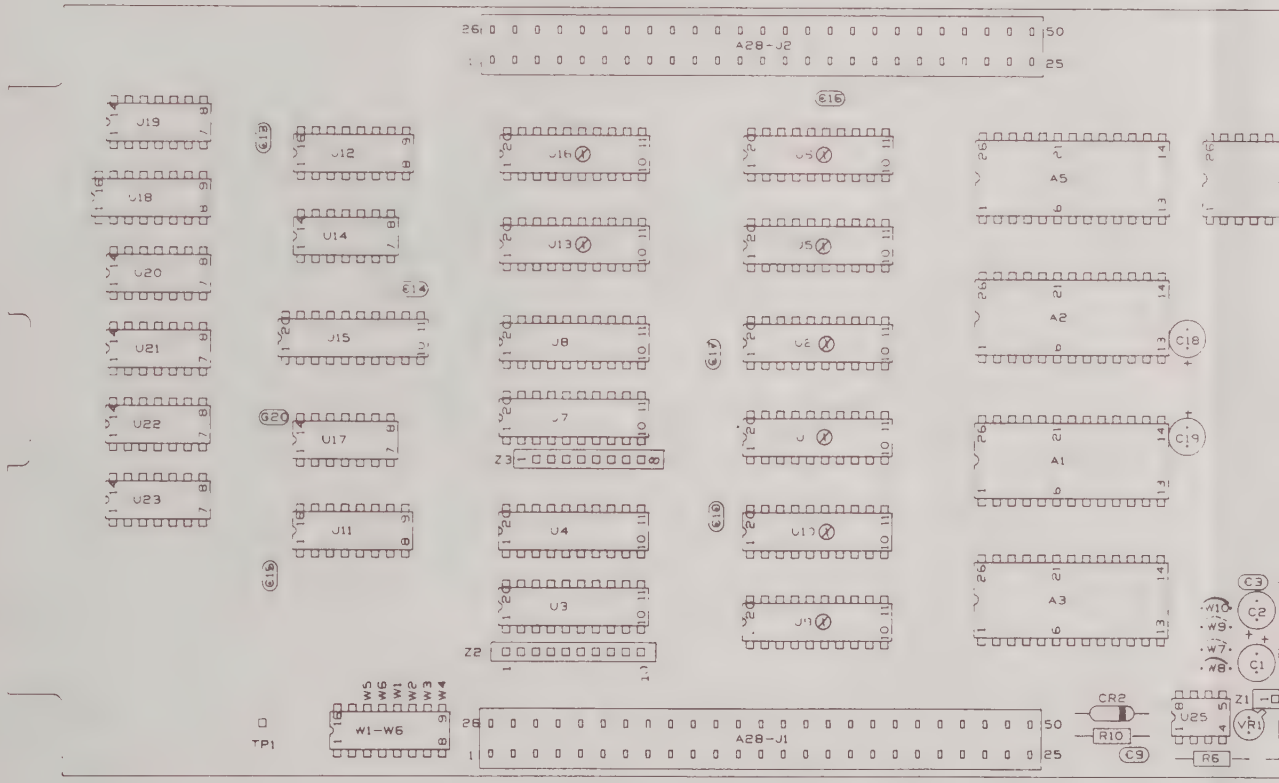
1. ALL RESISTORS ARE IN OHMS, 1/4W, 5%.
   ALL CAPACITORS ARE IN MICRO FARADS.



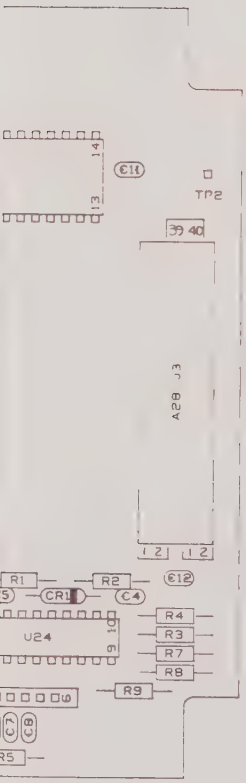
DES	DEVICE	PINS	GND	+5	QTY
U1	8086-1	40	1.20	40	1
U2 4	2732A-2	24	12	24	2
U3.5	HM6116P-3	24	12	24	2
U24.25	PAL 16L8	20	10	20	2
U23	PAL 16R8	20	10	20	1
U8.9	74SC245	20	10	20	2
U6	74LS374	20	10	20	1
U10,11,14,15	74LS373	20	10	20	4
U32.33	74LS273	20	10	20	2
U28	74LS193	16	8	16	1
U26	74F257	16	8	16	1
U12.13	74LS138	16	8	16	2
U21.30	74LS125	14	7	14	2
U20. 29.34	74LS74	14	7	14	3
U22.31	74LS32	14	7	14	2
U19	74LS27	14	7	14	1
U35	74LS10	14	7	14	1
U37	74F08	14	7	14	1
U36.38	74LS08	14	7	14	2
U17	74F04	14	7	14	1
U39	74LS02	14	7	14	1
U16.40	74LS00	14	7	14	2
U18	74LS38	14	7	14	1
Y1	K1116A	14	7	14	1
U7	74LS541	20	10	20	1
Z1	SIP RN 4.7K	8	—	1	1
U4.	PAL 16R4	20	10	20	1
	74F74	14	7	14	1

9000A-8086-1071
(2 of 2)

Figure 8-4. A27 Processor PCB Assembly (cont)

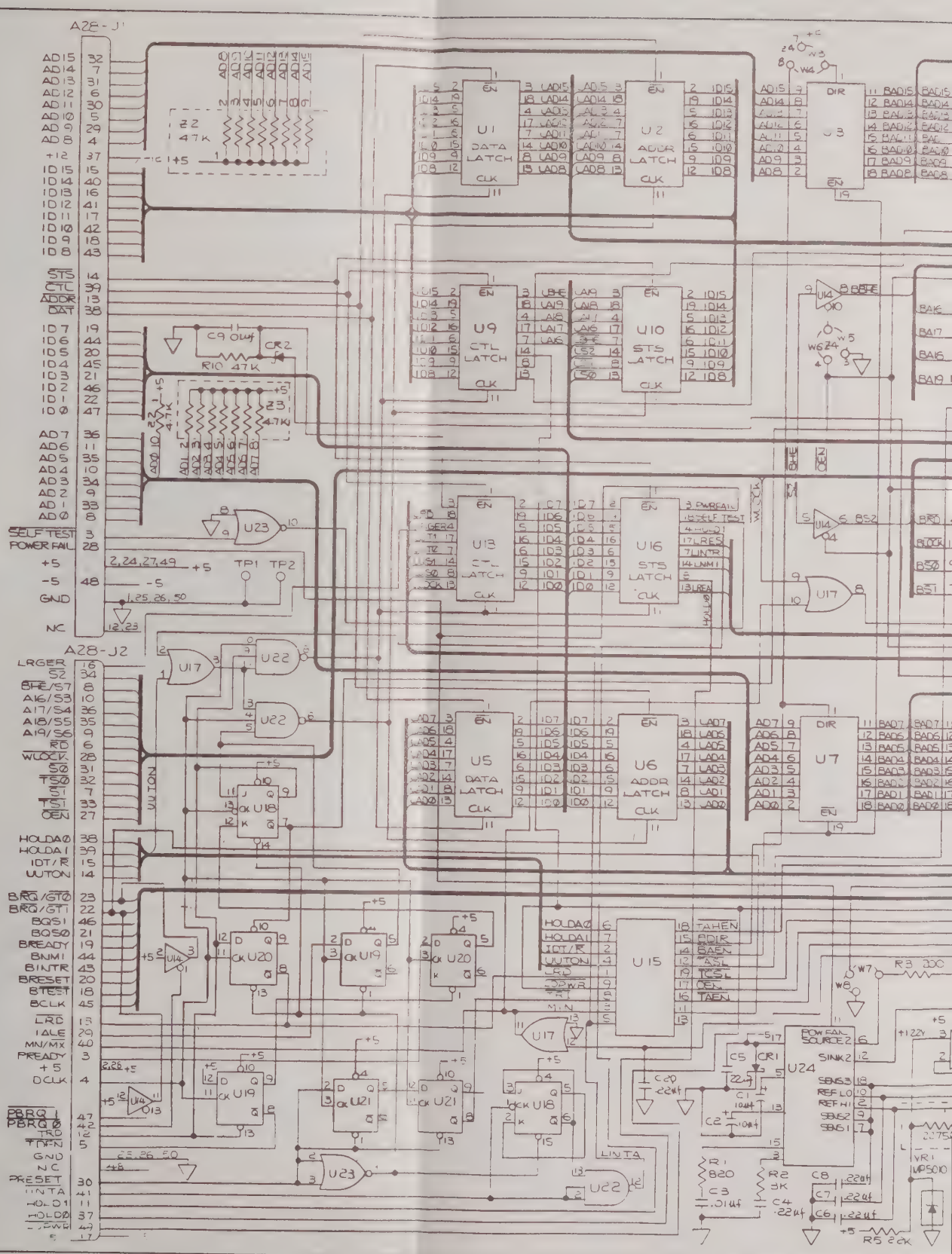


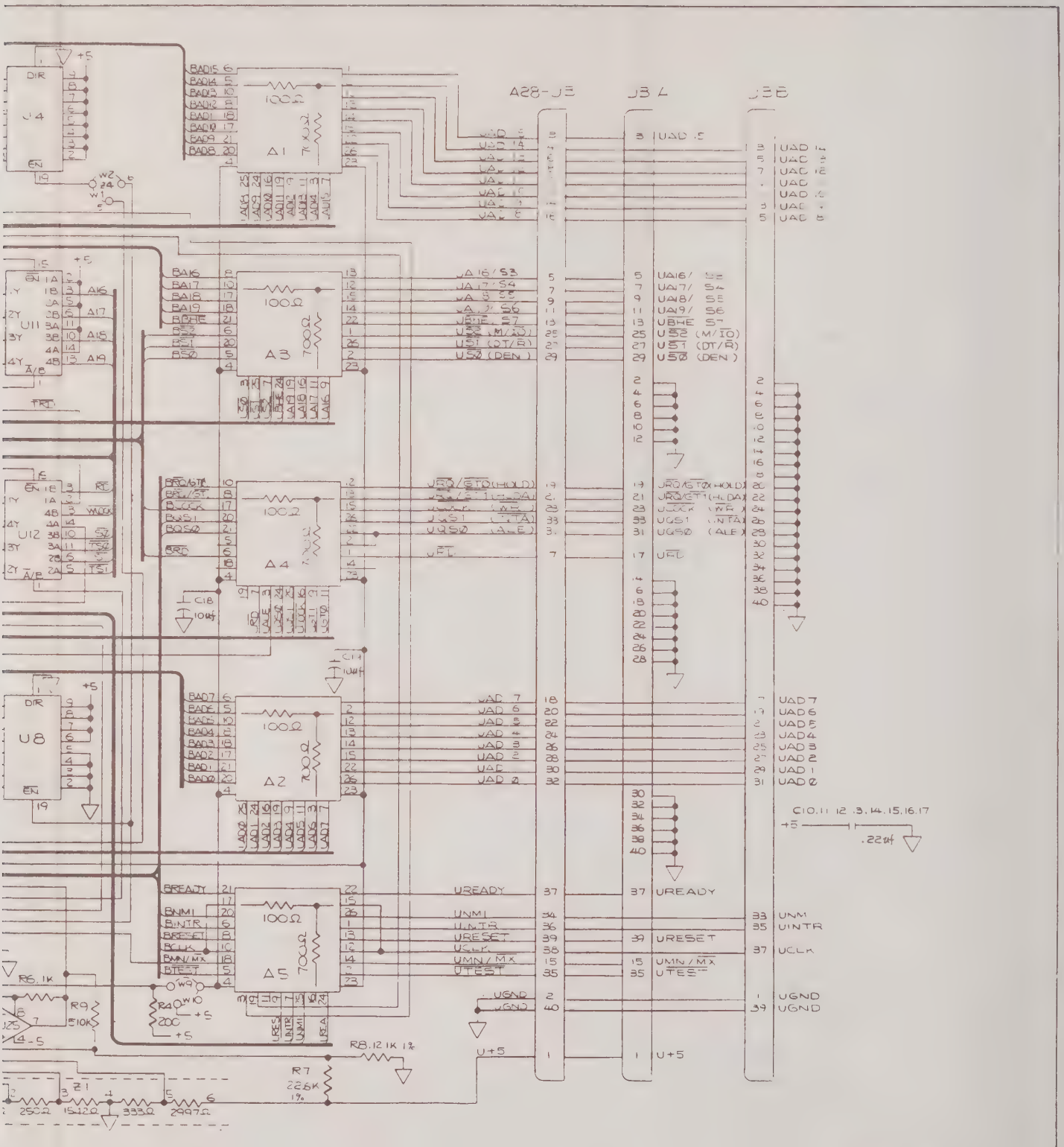
CAUTION
SUBJECT TO DAMAGE BY
STATIC ELECTRICITY



9000A-8086-1672

Figure 8-5. A28 Interface PCB Assembly


A2E - J1
A28 - J2
U1 DATA LATCH
U2 ADDR LATCH
U3 DIR
U5 DATA LATCH
U6 ADDR LATCH
U7 DIR
U9 CTL LATCH
U10 STS LATCH
U13 CTL LATCH
U16 STS LATCH
U15
U24
R10 47K
C9 .01uf
CR2
R23 4.7K
TP1 TP2
+5
-5
GND
NC
SELF TEST
POWER FAIL
LRGER
S2
BHE/S7
A16/S3
A17/S4
A18/S5
A19/S6
RD
WLOCK
S0
TS0
ST
TST
DEN
HOLDA0
HOLDA1
IDT/R
UUTON
BRQ/GT0
BRQ/GT1
BQS1
BQS0
BREADY
BNMI
BINTR
BRESET
BTEST
BCLK
LRD
IALE
MN/MX
PREADY
DCLK
PBRQ1
PBRQ0
TRD
TMEN
PRESET
INTA
HOLD1
HOLD0
LPEWR
U17
U22
U18
U20
U19
U21
U23
U14
U4
R1 820
R2 3K
R3 200
C1 10uf
C2 10uf
C3 .01uf
C4 .22uf
C5 .22uf
C6 .22uf
C7 .22uf
C8 .22uf
C20 .22uf
R5 2ck
VR1 LM5010



9000A-8086-1072
(1 of 2)

Figure 8-5. A28 Interface PCB Assembly (cont)

SPARE GATES

TABLE 1

DIP (& SIP) PKG DEFINITION

DES	DEVICE	PINS	GND	+5	QTY
A1,2,3,4,5	9000A-4HC4T	26	—	—	5
U1,2,5,6,9,10,13,16	74SC374	20	10	20	8
U4,8	74SC245	20	10	20	2
U11,12	74F257	16	8	16	2
U14	74LS125	14	7	14	1
U15	PAL16LS	20	10	20	1
U17	74LS32	14	7	14	1
U18	74LS112	16	8	16	1
U19	74S74	14	7	14	1
U20,21	74LS74	14	7	14	2
U22	74LS10	14	7	14	1
U23	74LS02	4	7	14	1
U24	9000A-8076	8	8,14,16	1,4	1
U25	LM311N	8	1	8	1
U3,7	74ALS245	20	10	20	2
Z1	54-4087T	6	4	—	1
Z2	SIP RN 4.7K	10	—	1	1
Z3	SIP RN 4.7K	8	—	1	1
Z4	PLCC JUMPER HEADER	16	—	—	1

TES: UNLESS OTHERWISE SPECIFIED

ALL RESISTORS ARE IN OHMS, 1/4, 5%.
ALL CAPACITORS ARE IN MICROFARADS.

TABLE 2

REFERENCE DESIGNATIONS

HEST NO	DES. NOT USED

TABLE 3

JUMPER WIRE TABLE

FOR 8086 CONFIGURATION CONNECT W2 W4, W6 ONLY
FOR 8088 CONFIGURATION CONNECT W1, W3, W5 ONLY

9000A-8086-1072
(2 of 2)

Figure 8-5. A28 Interface PCB Assembly (cont)

INDEX

The number following each index entry indicates the page number.

AD0 thru AD15, **3-1**
AD16/S3 thru AD19/S6, **3-1**
Address Mapping, **4-2**
Address Space Assignment, **4-2**
ALE Control Line, **3-4, 4-5**
Assembly Drawings, **7-1**

$\overline{BHE}$ Control Line, **3-2, 4-5**
Bit Assignments
  Control Lines, **4-5**
  Status Lines, **4-5**
Bus Test Default Addr., **4-16**
Byte Addresses, **4-2, -3**

Cascade Address, **4-17**
Code (CS) Register, **4-3**
Conn. to Troubleshooter, **2-1**
Conn. to UUT, **2-1**
Control Line Bit Assign., **4-5**
Control Section, **5-4**
CS Register, **4-3**

Data Memory, **4-2, -3**
Data (DS) Register, **4-3**
Defective Pod, **6-2**
$\overline{DEN}$ Control Line, **3-4, 4-5**
Disassembly of Pod, **6-12**
DS Register, **4-3**
DT/$\overline{R}$ Control Line, **3-4, 4-5**

Emulation (Run UUT), **4-14**
Enable Lines, **4-6**
Enhanced Self Test, **6-3 thru -6**
ES Register, **4-3**
External Features of Pod, **1-3**
Extra Data (ES) Register, **4-3**

Forcing Lines, **4-5, -7**

$\overline{GT0}$, $\overline{GT1}$ Control Lines, **4-5, -8**

HLDA Control Line, **3-3, 4-5, -8**
Hold Status Line, **3-3, 4-5, -6**
HOLD0 Status Line, **4-5 thru -7**
HOLD1 Status Line, **4-5 thru -7**

Inoperative Pod, **6-2, -7**
INT VECT Status Line, **4-5, -7**
Interface Pod
  Conn. to Troubleshooter, **2-1**
  Conn. to UUT, **2-1**
  Disassembly, **6-12**
  External Features, **1-3**
  Self Test (see Self Test)
  Specifications, **1-4**
Interface Section, **5-4**
Interrupt
  Disabling Interrupts, **4-17**
  Enabling Interrupts, **4-17**
  Forcing Inter. Ack., **4-17**
  Interrupt Handling, **4-16**
  Interrupt Lines, **4-8**
  Reading Interrupt Type, **4-17**
  Special Addresses, **4-18**
$\overline{INTA}$ Control Line, **3-4, 4-5, -8**
INTR Status Line, **3-4, 4-5 thru -8**
Installation in UUT, **2-1**
I/O
  Byte Accesses, **4-3**
  Word Accesses, **4-3**

Learn Oper. Default Addr., **4-16**
$\overline{Lock}$ Control Line, **3-3, 4-5, -8**
Low UUT Power Detection, **4-20**

Maximum Mode, **3-1, -5**
Memory Map, **4-3**
Microprocessor Data, **3-1 thru -5**
  Pin Assignments, **3-5**
  Signal Descriptions, **3-1**
Minimum Mode, **3-1, -5**
M/$\overline{IO}$ Control Line, **3-2, 4-5**
MN/$\overline{MX}$ Status Line, **3-2, 4-5**

NMI Status Line, **3-5, 4-5, -8**

Operating Info., **4-1 thru -20**

Parts List, **7-1**
Pin Assignments, **3-5**
Pod (see Interface Pod)
Power Fail Detection, **4-20**
Probe and Scope Sync. Modes, **4-18**
Protection Circuits, **5-4**
PWR FAIL Status Line, **4-5 thru -7**

QS0, QS1 Ctl. Lines, **3-3, 4-5, -8**
Quick RAM Test, **4-10, -12**
Quick Loop. Read/Write, **4-10, -12**
Quick ROM Test, **4-10, -13**

$\overline{RD}$ Control Line, **3-2, 4-5**
Ready Sts. Line, **3-5, 4-5 thru -7**
Reset Status Line, **3-5, 4-5, -7**
R/G Error Sts. Line, **4-5 thru -7**
$\overline{RQ}$/$\overline{GT0}$, $\overline{RQ}$/$\overline{GT1}$, **3-3, 4-5 thru -9**
Run UUT Mode
  Default Address, **4-14**
  Special Addresses, **4-15**
  Specifying Exec. Addr., **4-14**

Schematics, **8-1**
Segment Registers, **4-3**
Self Test
  Enhanced Self Test, **6-3 thru -6**
  Failure Codes
    Enhanced Self Test, **6-6**
    Standard Self Test, **6-3**
  How to Perform, **2-1**
  Socket, **1-3**
  Standard Self Test, **6-3, -6**
Special Addresses, **4-3**
Special Control Line, **4-5, -8**
Special Functions, **4-9**
Specifications, **1-4**
SS Register, **4-3**
Stack (SS) Register, **4-3**
Status Line Bit Assign., **4-5**
Synchronization Modes, **4-13**
$\overline{S2}$ thru $\overline{S0}$ Ctl. Lines, **3-2, 4-5**
S3 thru S7 Ctl. Lines, , **3-1, 4-5**

$\overline{Test}$ Status Line, **3-5, 4-5**
Theory of Operation
  Detailed Block Diag., **5-6**
  Detailed Description, **5-5**
  General Block Diag., **5-2**
  General Description, **5-1**
Timing and Control Section, **5-4**
Timing Diagram, **5-9**
Troubleshoot. Info., **6-1 thru -12**
  Defective Pod, **6-2**
  Inoperative Pod, **6-7**

User-Enableable Lines, **4-6**
User-Writable Control Lines, **4-8**

Word Addresses, **4-2, -3**
$\overline{WR}$ Control Line, **3-4, 4-5**
Writable Control Lines, **4-8**

www.manualsplus.com
1020860